

MXK-19/23 Configuration Guide

For software version 2.1

July 2010

Document Part Number: 830-01812-09

Zhone Technologies
@Zhone Way
7001 Oakport Street
Oakland, CA 94621
USA
510.777.7000
www.zhone.com
info@zhone.com

COPYRIGHT C2000-2010 Zhone Technologies, Inc. and its licensors. All rights reserved.

This publication is protected by copyright law. No part of this publication may be copied or distributed, transmitted, transcribed, stored in a retrieval system, or translated into any human or computer language in any form or by any means, electronic, mechanical, magnetic, manual or otherwise, or disclosed to third parties without the express written permission from Zhone Technologies, Inc.

Bitstorm, EtherXtend, IMACS, MALC, MXK, Raptor, SLMS, Z-Edge, Zhone, ZMS, zNID and the Zhone logo are trademarks of Zhone Technologies, Inc.

Zhone Technologies makes no representation or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability, non infringement, or fitness for a particular purpose. Further, Zhone Technologies reserves the right to revise this publication and to make changes from time to time in the contents hereof without obligation of Zhone Technologies to notify any person of such revision or changes.

TABLE OF CONTENTS

About This Guide	17
Style and notation conventions	17
Typographical conventions	18
Related documentation	18
Acronyms	18
Contacting Global Service and Support	20
Technical support	20
Hardware repair	21
Chapter 1 MXK	23
MXK overview	23
MXK features	27
Ethernet services	28
GPON	28
VoIP	28
MGCP	28
SIP	29
Redundancy on uplinks	30
Management	30
Data services	30
Rate Limiting	32
Chapter 2 MXK Operations, Administration, and Maintenance	33
MXK management interfaces	33
Log in to the serial (craft) port	33
Configure the MXK serial craft settings	34
Change the serial craft port settings	34
Log in and out of the system	35
MXK default configuration	35
Defaults overview	35
Monitoring the MXK through the serial craft port	36
Enable/disable temporary logging sessions	36
Configure management interfaces	37
Configure an out-of-band IP management interface	37
Configure an in-band management interface	38
Configure IP on a bridge for MXK management	38
Create the default route	41
Configure DNS resolver	41
Manage the MXK with the Web User Interface	43
Manage the MXK using Zhone Web User Interface	44
Disable the Web UI	45
Disable partial config sync traps to ZMS from the device	46
MXK card configuration	49
View uplink cards	50

View line cards	50
MXK card configuration.....	51
Add a card profile.....	51
Delete a card profile.....	52
MXK system administration.....	53
User account administration.....	54
Add users.....	54
Change default user passwords.....	55
Delete users.....	55
Delete the admin user account.....	55
Reset passwords.....	56
View chassis and system information.....	56
MXK basic system administration commands.....	57
Commands: new, list, show, get, update, delete.....	57
Commands: interface show, host show, bridge show.....	62
Commands: bridge stats.....	64
Commands: port show, port up, port down, port bounce.....	64
Navigate the MXK file system.....	65
Access the MXK file system.....	65
Download software files	66
Backup system configurations.....	67
SNTP.....	68
MXK Simple Network Management Protocol (SNMP).....	69
Create SNMP community names and access profiles.....	69
Configure traps.....	70
Use log files to monitor the system.....	71
Overview.....	71
Enable/disable logging.....	72
Log message format.....	72
Modify logging levels.....	74
Non-persistent log messages.....	75
Persistent log messages.....	77
Example log messages.....	77
Log filter command.....	78
Send messages to a syslog server.....	79
Specify different log formats for system and syslog messages.....	80
Map subscriber information to a port description field on the MXK	82
Port description rules.....	82
Add, modify, list, and delete a port description.....	82
Add a port description to a port.....	83
Add a port description to a GPON port.....	83
Add a port description to a bridge.....	84
Add a port description to a bond group.....	85
Modify a port description.....	85
Port description list.....	86
Port description delete.....	86
Search port descriptions.....	86
MXK security.....	87
MXK security (SSH, SFTP, and HTTPS).....	87

	Enable security on the MXK.....	87
	DSA and RSA keys.....	89
	Tested MXK SSH clients.....	90
	Encryption-key commands.....	90
	Port access security.....	91
	Radius support.....	94
	MXK clocking	98
	View clocking source on the MXK.....	98
	Set the TAC card as clocking source.....	99
	MXK alarms	100
	Alarm manager.....	100
	Alarm suppression.....	101
Chapter 3	MXK IP Service Level Agreement (IPSLA)	105
	IPSLA configuration	105
Chapter 4	IP Configuration	115
	IP overview	115
	IP services.....	116
	IP protocols.....	116
	DNS.....	116
	DHCP.....	116
	Routing Information Protocol (RIP).....	117
	IP Quality of Service (QoS).....	118
	ToS, CoS, and sCoS on IP interfaces.....	118
	ToS fields in an IP header.....	119
	CoS fields in a VLAN header.....	119
	ToS, CoS, sCoS parameters.....	120
	802.1p priority queues.....	122
	Floating IP interfaces.....	123
	Routing overview	123
	Host-based (unnumbered) routing overview.....	124
	Network-based (numbered) routing overview.....	125
	DHCP overview	126
	MXK DHCP server support.....	126
	DHCP server profiles and scope.....	127
	DHCP server options.....	127
	DHCP server subnet options.....	128
	MXK DHCP relay.....	130
	MXK routing configurations	132
	Host-based routing.....	132
	Static host-based routing without DHCP	133
	Delete hosts.....	135
	Static and dynamic host configuration with the same subnet.....	136
	Host-based routing with the MXK as a local DHCP server.....	137

Host-based routing with the MXK as a local DHCP server to provide DNS and bootp services.....	140
Host-based routing with an external DHCP server.....	144
Host-based routing with multiple dhcp-relay agents and one DHCP server..	147
Host-based routing with an external DHCP server and an alternate DHCP server with dhcp-relay agent	150
Network-based routing.....	152
Network-based routing without DHCP	153
Network-based routing with the MXK as local DHCP server.....	154
Network-based routing with an external DHCP server.....	156
Host-based routing for triple-play services on Ethernet.....	158
Host-based routing for triple-play services on GPON.....	163
Static routes.....	168
Add routes.....	168
IP fallback route.....	169
RIP configuration.....	171
IP administrative procedures	171
Modify profiles created by host/interface add commands.....	172
Display hosts.....	172
Display interfaces.....	173
Display routing information.....	173
Displaying the routing table.....	173
Displaying RIP information.....	173
Delete hosts.....	174
Delete interfaces.....	174
Delete routes.....	174
DHCP logging.....	175
Enable DHCP logging.....	175
DHCP server log messages.....	176
View client leases.....	176
IP statistics commands.....	178
CPE Manager	181
Verifying CPE Manager.....	184
Additional information about CPE manager.....	185
Finding the local IP address and CPE base port.....	186
Web UI cut-through for EtherXtend devices.....	187
 Chapter 5 Bridging Configuration	 189
Overview	190
Terminology and concepts	191
Physical port.....	192
Physical interface.....	192
Logical interface.....	193
Bridges and bridge interfaces.....	193
VLANs and SLANs, untagged, tagged and stagged.....	193
Upstream and downstream.....	196
Broadcast, multicast, and unicast.....	198

SLMS bridge types	198
Transparent LAN services.....	200
Configure a TLS bridge.....	202
Asymmetric bridges.....	203
Custom ARP.....	205
Configure an uplink bridge.....	206
Intralinked SLMS devices.....	207
Tagging operations	211
Overview.....	211
Common tagging operation scenarios.....	214
Tagging options.....	221
Common bridge commands	226
bridge add command.....	226
Verifying bridge interface settings.....	226
Bridge add and bridge-path add.....	227
Settings for asymmetric bridges	229
Settings for symmetric bridges	230
Bridge configuration with DHCP relay	232
Shaping Traffic: Class of Service Queuing	235
Configuring Class of Service.....	236
PPPoA - PPPoE interworking	237
Bridges with packet rule records	240
Mechanism for multiple interface ingress filters.....	240
Configure packet-rule-records.....	242
Destination MAC swapping.....	243
Bandwidth limiting by port and service.....	245
Rate limiting overview.....	245
Color blind rate limiting.....	248
Configure color blind policing.....	248
Color aware rate limiting.....	252
Configure color aware policing.....	252
DHCP on bridge packet rules (DHCP relay, Option 82, and Forbid OUI).....	253
Filtering access based on organizational unique indentifier (OUI).....	255
PPPoE with intermediate agent	256
ADSL2+ line rate on PPPoE-IA signaling.....	257
PPPoA - PPPoE interworking.....	258
Advanced bridging topics	260
Bridges with IGMP.....	260
Verifying bridge settings.....	262
“Denial of Service” prevention.....	264
Rapid Spanning Tree Protocol (RSTP).....	264
RSTP port role.....	265
RSTP port state.....	266
RSTP on uplinks.....	266
RSTP rlinks.....	269
Bridging differences between the MALC and MXK.....	274

MXK bridging configurations	274
Configure a tagged uplink bridge with VLAN ID	274
Configure tagged or untagged downlink bridges with VLAN IDs	276
Untagged downlink bridges on Active Ethernet	276
Configure tagged downlink bridges on Active Ethernet	277
Delete uplink and Active Ethernet downlink bridges	278
Configure uplink and downlink bridges on GPON for triple-play services	279
Configure bridges using Q-in-Q (VLAN IDs and SLAN IDs)	282
Q-in-Q parameters	282
Turn off Q-in-Q for the entire MXK system	283
Q-in-Q bridging configurations on Active Ethernet	284
Configure bridges using VLAN 0	290
MXK bridging configuration with VLAN 0 on uplink and downlink bridges	290
MXK bridging configuration with VLAN 0 on TLS bridges for multi-point connections	293
MXK bridging configuration with VLAN 0 on tagged intralinks	296
MXK bridging configuration with VLAN 0 on staged intralinks	298
Configure TLS bridges	300
Configure TLS wire bridges	301
TLS bridge parameters floodUnknown and floodMulticast	302
floodUnknown parameter	302
floodMulticast parameter	303
Configure link aggregation bridges	304
Create a uplink bridge with link aggregation	305
Create a TLS bridge with link aggregation	305
Configure bridge loop issue prevention	306
Dynamic IP filtering on a bridge (Secure DHCP)	307
Broadcast suppression	309
Administrative commands	310
Bridge delete command	311
Bridge show/showall commands	311
Statistics on demand	312
 Chapter 6 Video Configuration	 315
MXK routed video	315
Configure routed video connections on the MXK	316
Routed video configuration overview	316
Configure host-based routing for video with local DHCP	316
Configure host-based routing for video with dhcp-relay agent(s)	322
MXK bridged video	331
Configure bridged video on the MXK	332
Bridged video connection overview	333
Configure a video connection on the MXK	333
IGMP snooping with proxy reporting	340
IGMP snooping overview	340
Join requests	340

	Leave requests.....	341
	IGMP snooping with proxy configuration.....	341
	IGMPv3 messages respond for STBs.....	346
	IGMP bridging statistics	346
Chapter 7	Link Aggregation Configuration	349
	Link aggregation overview.....	349
	Link aggregation and LACP.....	350
	Link aggregation modes.....	350
	Link resiliency.....	351
	Ethernet uplink ports available for link aggregation.....	351
	Ethernet line card ports available for link aggregation.....	352
	Configure link aggregation on Ethernet uplink ports	353
	Configure Ethernet uplink ports for manual link aggregation.....	354
	Configure Ethernet uplink ports for LACP.....	355
	Delete a link aggregation group.....	356
	Configure link aggregation on Ethernet line card ports	357
	Configure Ethernet line card ports for manual link aggregation.....	357
	Configure line card Ethernet ports for LACP.....	359
	Delete a link aggregation group.....	360
	lacp command	360
	Configure link aggregation bridges	361
	Configure link aggregation uplink bridges.....	361
	Configure link aggregation line card bridges.....	362
Chapter 8	MXK 100/1000 Ethernet and 10 GE Uplink Cards	365
	100/1000 Ethernet and 10 GE uplink cards	366
	100/1000 Ethernet and 10 GE uplink card overview.....	366
	Uplink card specifications.....	367
	Redundant uplink card configuration.....	369
	Disable Tx power on the uplink standby card.....	372
	View additional card and system information.....	373
	Displaying and updating Ethernet interfaces.....	374
	EAPS	376
	Advanced EAPs topics.....	379
	Common EAPs scenarios.....	380
	Small form factor pluggables	383
	Uplink card pinouts	383
	Serial (craft) port pinouts.....	384
	Ethernet port pinouts.....	384

Chapter 9	MXK GPON Cards	387
	GPON cards	388
	GPON card overview	388
	GPON card specifications	389
	GPON card configuration	390
	View additional card and system information	391
	GPON configuration	392
	Smart OMCI	392
	OMCI overview	392
	Smart OMCI overview	393
	Configure Smart OMCI on ONU	395
	Delete OMCI profile	410
	Import and export OMCI profile	413
	GPON type B redundancy	417
	Switchover from active to standby GPON port	423
	GPON redundancy configuration limitations	424
	MXK GPON using the Reg ID for provisioning	424
	Configuring Reg ID	425
	GPON ONU serial number format (Hexadecimal or Decimal)	426
	Associate a vendor ID and a serial number with an ONU when activating the ONU	428
	GPON extended reach	428
	Recommendations for extended reach	429
	Commands to enable extended reach	430
	Received Signal Strength Indication (RSSI) and Digital Diagnostic Monitoring (DDM)	431
	Manage ONU with OMCI	434
	View OMCI configuration states and alarms on ONU	434
	Retrieve status of GPON subscriber ports	436
	Administration of GPON subscriber ports	436
	Configurable speed of GPON subscriber ports	437
	Configurable Transmit Gain, Receive Gain, and Fax Mode parameters for POTS ports	437
	Manual upgrade on an ONU	439
	Auto upgrade on an ONU	443
	View the ONU upgrade status	447
	Reboot an ONU	449
	Re-synchronize an ONU	449
	Add new services for subscriber without affecting existing services	449
	Retrieve alarm information on an ONU	449
	View or change trap reporting status on an ONU	450
	Dynamic GEM ports	451
	Create a GEM port with a GPON traffic profile	451
	View the GEM port-related information	458
	Bandwidth allocation	459
	Configure GPON traffic profile	459
	Dynamic Bandwidth Allocation (DBA)	468

OMCI Statistics	471
PON Statistics	474
Chapter 10 MXK 20-Port Active Ethernet Card.....	487
20-port Active Ethernet card	488
Active Ethernet card overview.....	488
Active Ethernet card specifications.....	489
Active Ethernet 20 port card configuration.....	489
View additional card and system information.....	491
Small form factor pluggables	492
Displaying and updating Ethernet interfaces	492
Chapter 11 MXK 20-Port Single Slot Active Ethernet Card.....	495
20-port single slot Active Ethernet card	496
Active Ethernet card overview.....	496
Active Ethernet card specifications.....	497
Active Ethernet 20-port single slot card configuration.....	497
View additional card and system information.....	499
Small form factor pluggables	500
Display and update Ethernet interfaces	500
Chapter 12 MXK ADSL2+ Bond Cards	503
ADSL2+ bond cards	504
ADSL2+ card overview.....	504
ADSL2+ card specifications.....	506
ADSL2+ bond card configuration.....	508
View additional card information.....	510
ADSL on the MXK	510
ADSL overview.....	511
ADSL2+ transmission modes.....	511
ADSL2+ rate adaptation.....	512
Advanced ADSL2+ configurations on the MXK.....	512
Fine tuning ADSL2+ video performance.....	513
Seamless Rate Adaptation	516
Transport mode: fast or interleaved.....	518
Configure ADSL2+ interfaces.....	521
ADSL2+ interface overview.....	521
View adsl-profile parameter defaults.....	522
View adsl-co-profile parameter defaults.....	525
View adsl-cpe-profile parameter defaults.....	530
Upstream and downstream tone ranges.....	536
Configure ADSL2+ profiles for Annex M in fast mode.....	536
Configure ADSL2+ profiles for Annex M in interleaved mode.....	538

Configure ADSL2+ profiles for G.lite.....	541
Configure ADSL2+ profiles to cap train rates.....	543
Configure ADSL2+ S=1/2.....	548
Configure Broadcom Phy-R™ parameters.....	553
ADSL2+ statistics	555
ADSL2+ bonding.....	566
ATM on the ADSL2+ POTS line card	569
ATM data.....	569
VPI and VCI ranges.....	569
Service categories.....	570
Constant Bit Rate (CBR).....	570
Non-real-time variable bit rate (nrt-VBR).....	570
Real-time variable bit rate (rt-VBR).....	570
Unspecified bit rate (UBR).....	570
Traffic descriptors.....	570
Traffic descriptor parameters.....	571
ATM sample configurations.....	571
ATM traffic descriptor example for data.....	571
ATM traffic descriptor example for video.....	572
ATM statistics.....	572
POTS to Voice over IP (VoIP) configuration	572
VoIP overview.....	572
Basic VoIP configuration.....	573
Configure a IP interface for voice traffic.....	573
SIP server configuration.....	574
SIP dial plan configuration.....	576
SIP PLAR server configuration.....	578
MGCP configuration.....	579
MEGACO (H.248 configuration).....	581
Additional VoIP features.....	584
Advanced VoIP configuration.....	587
ToS, CoS, and sCoS on IP interfaces.....	587
ToS fields in an IP header.....	587
CoS and Tos for all IP packets.....	589
ToS configuration to add priority to voice packets.....	590
Emergency Stand Alone (ESA) SIP support	592
Configuring VoIP ESA clusters.....	593
Configuring ESA for 911 calls.....	595
Verifying ESA.....	595
ADSL2+ cable and port pinouts.....	595
ADSL2+ card port pinouts.....	595
ADSL-48 card pinouts.....	596
ADSL2+ cable pinouts.....	600
ADSL-48 to dual 50-pin connector cable	600
48-port ADSL to dual 50-pin connector cables.....	604
Variations of 48-port ADSL to dual 50-pin connector cables.....	605
ADSL2+ testing (SELT/DELT) on the MXK	606

SEL T (Single-End Loop Test).....	606
DELT (Dual-End Loop Test).....	611
Chapter 13 MXK 24-Port VDSL2 Card	617
VDSL2 24-port single slot card	618
VDSL2 card overview.....	618
VDSL2 card specifications.....	619
VDSL2 24-port card configuration.....	619
View additional card information.....	622
VDSL2 interfaces	622
VDSL interface profiles.....	623
vdsl-config default parameters.....	623
vdsl-co-config default parameters.....	627
View vdsl-cpe-config profile default parameters.....	630
Configure VDSL2 profiles to cap train rates.....	634
Configure Broadcom Phy-R™	634
VDSL2 statistics	636
View VDSL2 statistics.....	636
View VDSL2 statistics with the -v variable.....	637
Clear VDSL2 counters	638
VDSL statistics parameters.....	639
VDSL2 24 port card pinouts	645
Chapter 14 MXK EFM SHDSL Cards	647
EFM SHDSL cards	648
EFM SHDSL card overview.....	648
EFM SHDSL card specifications.....	649
EFM SHDSL-24 card configuration.....	650
Enter a card-profile for the card.....	650
Set wetting current.....	652
Switch clocking source.....	652
MXK EFM SHDSL bonding overview	653
G. SHDSL bond group configuration	654
Conditions and limitations for cross-card bonding.....	654
Bond group bandwidth specifications.....	655
Bond group configuration.....	655
EFM auto bonding	655
EFM manual bond groups.....	658
Create bond groups on one card.....	658
View bond groups.....	659
Change bond group type.....	661
Move bond group members.....	662
Delete bond groups.....	662
Cross-card bonding.....	663
SHDSL error monitoring.....	664

SHDSL error monitoring statistics.....	664
SHDSL error monitoring fields.....	664
Configure the pme-profile	665
Configure automatic baud rate adaption and fixed rate settings.....	666
Configure auto-negotiate or specific data rate.....	667
Configure constellation for a TCPAM setting.....	668
Set a region.....	670
SNR monitoring for bonded G.SHDSL lines	671
SNR monitoring for the MXK	671
SNR monitoring for the MXK overview.....	671
Current condition SNR maximum threshold.....	672
Current condition minimum SNR threshold.....	672
MXK SNR monitoring pme-profile parameters.....	673
Usage for SNR pme-profile and efm-port parameters.....	674
MXK SNR monitoring configuration.....	675
Set SNR for target current condition or target worst case mode.....	675
Set MXK time and day.....	676
Set SNR monitoring from the CLI.....	676
View SNR monitoring statistics.....	679
Set SNR monitoring in the pme-profile	680
Configure SNR crossing traps.....	683
Verify SNR monitoring is enabled/disabled.....	683
G. SHDSL SNR monitoring example.....	684
Disable SNR monitoring.....	689
SHDSL error monitoring	689
SHDSL error monitoring statistics.....	689
SHDSL error monitoring fields.....	690
SHDSL statistics	691
Bond group statistics and port statistics	695
View port statistics	695
View bond group statistics.....	696
802.3ah EFM OAM	696
MXK-EFM-SHDSL-24 pinouts	699
Power and data connections for SHDSL CPE devices	700
Deliver power and data to the CPE	700
Enable power on the SHDSL line.....	702
MTAC testing	703
 Chapter 15 MXK Metallic Test Access (MTAC) Cards	 705
MTAC cards.....	706
MTAC card overview.....	706
MTAC card specifications.....	708
Connectors on the MTAC cards.....	709
Metallic loop testing.....	709
Internal look out line test.....	710

Cards supporting look-out test access.....	710
Ring generator.....	710
Configure MTAC cards.....	711
Perform line testing using MTAC cards with external testing set	713
Connect the external test set to the MTAC card.....	714
Connect the test measurement device to the metallic test access port.....	715
Connect a console to the external test set control port.....	717
Perform internal line testing with MTAC/RING-ENH card	718
Working with the MTAC line test command.....	718
Test IDs.....	720
Metallic loop tests.....	722
3 elements capacitance test.....	723
3 elements insulation resistance test.....	724
DC feed self-test.....	726
DC loop resistance test.....	726
Distance to open test.....	727
DTMF and pulse digit measurement test.....	728
Foreign AC currents test.....	729
Foreign DC voltage test.....	730
Foreign AC voltage test.....	730
Howler test.....	731
Metering self test.....	732
Noise test.....	732
On-Off hook transition test.....	733
Loop and battery condition test.....	733
Receiver off-hook test.....	734
Ringer equivalency number test.....	735
Ringing self test.....	735
Ringing monitor test.....	736
Tone generation test.....	736
Trans-hybrid loss test.....	737
Transmission self test.....	738
Troubleshooting with metallic loop tests	738
Auto-calibration.....	741
Lookout block diagram.....	742
Configure external alarms	742
Configure an external clock.....	743
Connect an external ring source	744
MTAC cards pinouts	746
External ring generator input port pinouts.....	747
External alarm sense pinouts.....	747
Examples of alarms with specific pinouts.....	750
Metallic test access port pinouts.....	753
External test set control port pinouts.....	754
External clock input port pinouts.....	756

Chapter 15 Small Form Factor Pluggable (SFP) Connectors.....757

Small form factor pluggables (SFPs) 757

 SFP and XFP overview.....757

 Supported SFPs.....758

 SFP specifications.....758

 Bi-directional SFPs.....760

 XFP specifications.....760

 GPON Class B+ SFPs.....761

Insert and remove a fiber connection and an SFP 762

Insert and remove a dual bi-directional SFP and fiber connector 763

Index.....765

ABOUT THIS GUIDE

This guide is intended for use by installation technicians and system and network administrators. It explains how to configure the MXK, provision uplink and line cards, create IP interfaces, configure bridges, and other system administration and networking tasks.

This chapter describes:

- [Style and notation conventions, page17](#)
- [Typographical conventions, page18](#)
- [Related documentation, page18](#)
- [Acronyms, page18](#)
- [Contacting Global Service and Support, page20](#)

Style and notation conventions

The following conventions are used in this document to alert users to information that is instructional, warns of potential damage to system equipment or data, and warns of potential injury or death. Carefully read and follow the instructions included in this document.

Caution: A caution alerts users to conditions or actions that could damage equipment or data.



Note: A note provides important supplemental or amplified information.

Tip: A tip provides additional information that enables users to more readily complete their tasks.

WARNING! A warning alerts users to conditions or actions that could lead to injury or death.

WARNING! A warning with this icon alerts users to conditions or actions that could lead to injury caused by a laser.

Typographical conventions

[Table 1](#) describes the typographical styles that this guide uses to represent specific types of information.

Table 1: Typographical styles

Bold	Used for names of buttons, dialog boxes, icons, menus and profiles when placed in body text, and property pages (or sheets). Also used for commands, options, parameters in body text, and user input in body text.
Fixed	Used in code examples for computer output, file names, path names, and the contents of online files or directories.
Fixed Bold	Used in configuration examples for text entered by users.
<i>Italic</i>	Used for book titles, chapter titles, file path names, notes in body text requiring special attention, section titles, emphasized terms, and variables.
PLAIN UPPER CASE	Used for environment variables.
Command Syntax	Brackets [] indicate optional syntax. Vertical bar indicates the OR symbol.

Related documentation

Refer to the following documents for additional information:

MXK Hardware Installation Guide — explains how to configure bridging, GPON, link aggregation, and other configuration tasks.

Zhone CLI Reference Guide — explains how to use the Zhone command line interface (CLI) and describes the system commands and parameters.

Refer to the release notes for software installation information and for changes in features and functionality of the product (if any).

Acronyms

[Table 2](#) provides a description of the acronyms that are related to Zhone products and may be found in this manual.

Table 2: Acronyms and their descriptions

Acronym	Description
ARP	Address resolution protocol
ATM	Asynchronous Transfer Mode
IAD	Integrated access device
MALC	Multi-access line concentrator
MIB	Management information bases
OLT	Optical line terminal
ONT	Optical network terminal
ONU	Optical network unit
PBX	Private branch exchange
POTS	Plain old telephone service
RIP	Routing Information Protocol
SFP	Small form factor pluggable
SLMS	Single Line Multi-Service
SNMP	Simple Network Management Protocol
TAC	Test Access Card
TFTP	Trivial File Transfer Protocol
XFP	10 Gigabit Ethernet small form factor pluggable
ZMS	Zhone Management System

Contacting Global Service and Support

If your product is under warranty (typically one year from date of purchase) or you have a valid service contract, you can contact Global Service and Support (GSS) for questions about this or other Zhone products, or for Technical Support or Hardware Repairs.

Before contacting GSS, make sure you have the following information:

- Zhone product you are using
- System configuration
- Software version running on the system
- Description of the issue
- Your contact information

If your product is not under warranty or you do not have a valid service contract, please contact GSS or your local sales representative to get a quote on a service plan. You can view the options on our web site at

<http://www.zhone.com/support/services/warranty>.

Technical support

The Technical Assistance Center (TAC) is available with experienced support engineers who can handle questions, assist with service requests, and help troubleshoot systems.

Hours of operation	Monday - Friday, 8 a.m. to 5 p.m, Pacific (excluding U.S. holidays)
Telephone (North America)	877-ZHONE20 (877-946-6320)
Telephone (International)	510-777-7133
E-mail	support@zhone.com
The Web is also available 24 x 7 to submit and track Service Requests (SR's)	www.zhone.com/support

If you purchased the product from an authorized dealer, distributor, Value Added Reseller (VAR), or third party, contact that supplier for technical assistance and warranty support.

Hardware repair

If the product malfunctions, all repairs must be authorized by Zhone with a Return Merchandise Authorization (RMA) and performed by the manufacturer or a Zhone-authorized agent. It is the responsibility of users requiring service to report the need for repair to GSS as follows:

- Complete the RMA Request form (<http://www.zhone.com/account/sr/submit.cgi>) or contact Zhone Support via phone or email:
Hours of operation: Monday Friday, 6:30am-5:00pm (Pacific Time)
E-mail: support@zhone.com (preferred)
Phone: 877-946-6320 or 510-777-7133, prompt #3, #2
- Provide the part numbers and serial numbers of the products to be repaired.
- All product lines ship with a minimum one year standard warranty (may vary by contract).
- Zhone will verify the warranty and provide the customer with a repair quote for anything that is not under warranty. Zhone requires a purchase order or credit card for out of warranty fees.

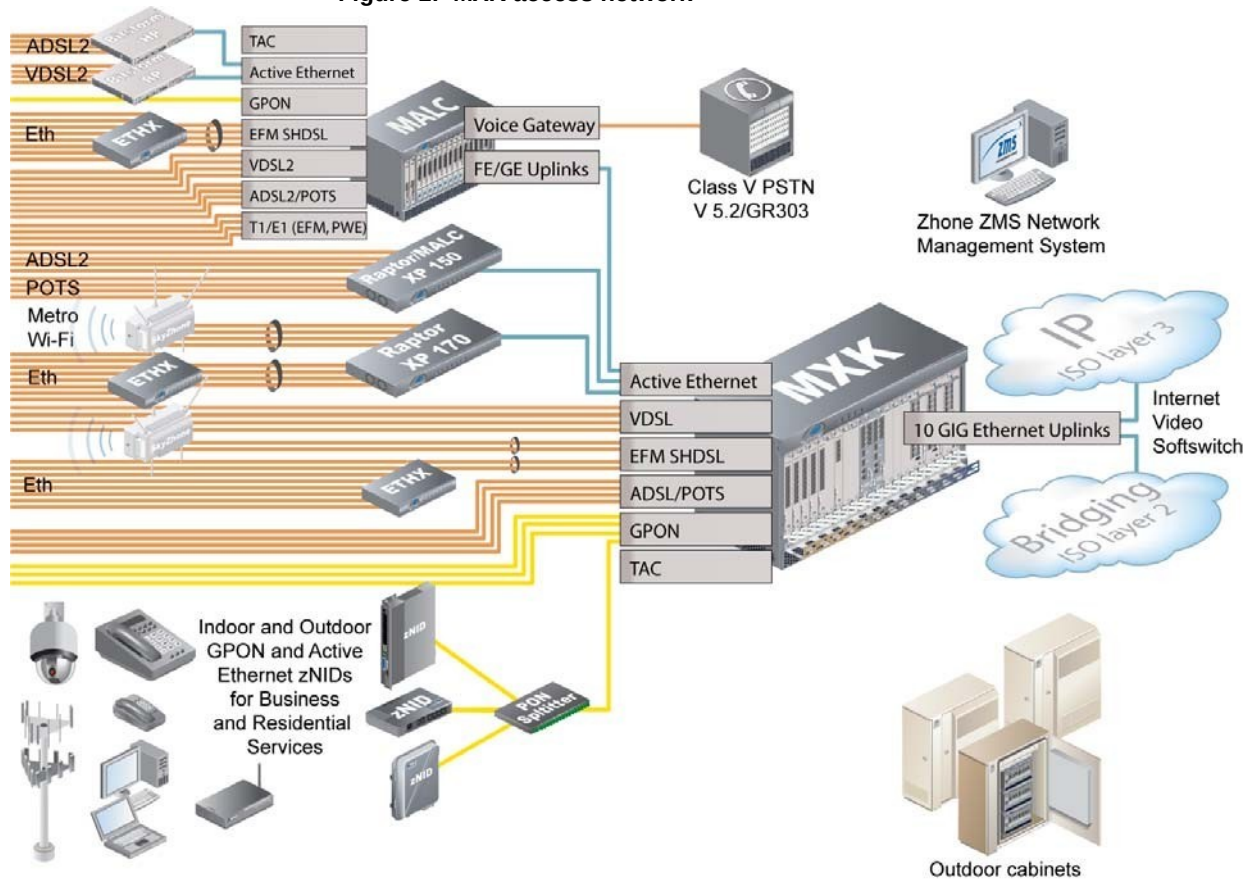
The logo for MXK 1, featuring the call letters "MXK" in a bold, black, sans-serif font above a large, light gray numeral "1". The "1" has a diagonal cut on its upper left side.

- MXK overview, page23
- MXK features, page27

MXK overview

The MXK platform provides high-density subscriber access concentration in the Zhong Single Line Multi-Service (SLMS) architecture. Figure 1 describes a typical MXK scenario.

Figure 1: MXK access network

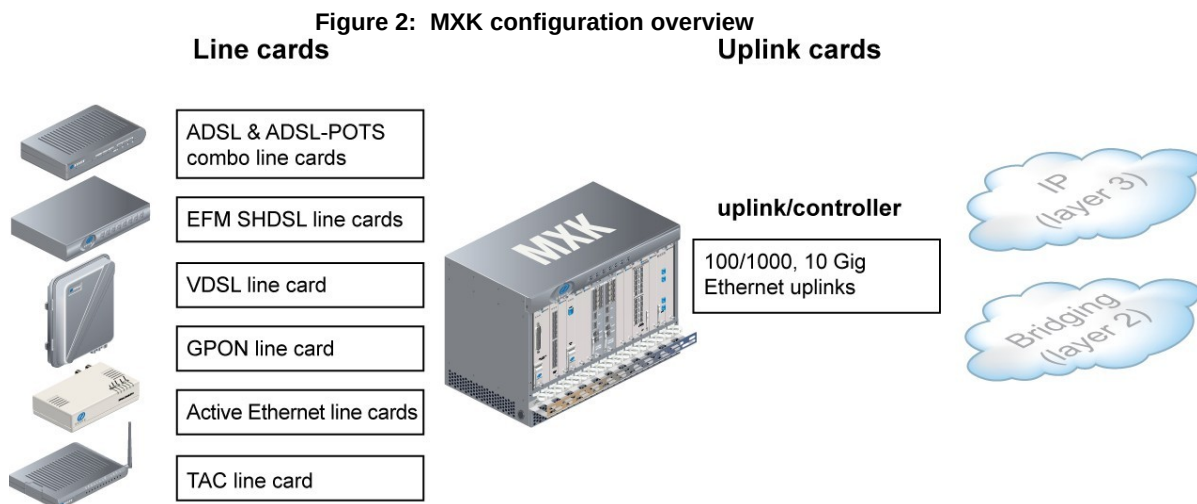


The MXK, in conjunction with zNIDs, provides a complete end-to-end access solution for fiber deployments (GPON and Active Ethernet) that provide triple-play services to subscribers. zNIDs at customer sites extend network intelligence all the way to subscribers with the ability to fine-tune performance.

The MXK provides advanced Quality of Service (QoS), Ethernet, and IP features that have been proven in carrier networks around the world. The MXK also includes improved cable and fiber management as well as IP and Ethernet packet forwarding.

MXK redundant uplinks are the primary communication channel between subscribers and upstream networking devices. The MXK aggregates local loop traffic from a variety of media and sends it to an upstream device, such as an IP router. The MXK supports GPON, Active Ethernet, EFM SHDSL, and ADSL edge connection technologies along with 100/1000 Ethernet and 10 Gigabit (GE) uplinks.

The redundant Ethernet uplinks on the MXK enable network providers to provision all classes of services in a single platform and leverage the existing copper infrastructure going to the Digital Loop Carrier (DLC) locations. [Figure 2](#) suggests the different types of network technologies the MXK supports.



The MXK can be deployed in Central Office environments or outdoor controlled environmental vaults for remote terminal applications. The MXK is intended for restricted access locations only.

MXK cards are divided into the following two types:

- There are three uplink cards for the MXK:
 - MXK MXK-UPLINK-2X10GE-8X1GE

Provides high-speed Gigabit Ethernet interfaces with active/standby redundancy. This uplink card consists of two 10 GE and eight 100/1000 Ethernet interfaces.

- MXK MXK-UPLINK-8X1G
Provides high-speed Gigabit Ethernet interfaces with active/standby redundancy. This uplink card consists of eight 100/1000 Ethernet interfaces.
- MXK-UPLINK-4X1GE
Provides high-speed Gigabit Ethernet interfaces with active/standby redundancy. This uplink card consists of four 100/1000 Ethernet interfaces and supports all line cards.
- MXK-UPLINK-4X1GE-CU
Provides high-speed Gigabit Ethernet interfaces with active/standby redundancy. This uplink card consists of four 100/1000 Ethernet interfaces and supports only copper line cards.
- GPON, Active Ethernet, ADSL2+, VDSL2, and EFM SHDSL line cards provide customer interfaces for services like IP TV, VoIP, and data.

The MXK line cards are:

- Active Ethernet

MXK-AEX20-FE/GE-2S

A two slot card that supports Ethernet traffic over 20 ports that provide either 10/100/1000 Base-T, fiber 100FX or 1 Gigabit Ethernet interfaces to support distances as high as 80km depending on the SFPs used.

MXK-AEX20-FE/GE

A slot card that supports Ethernet traffic over 10 ports that provide either 10/100/1000 Base-T, fiber 100FX or 1 Gigabit Ethernet interfaces to support distances as high as 80km depending on the SFPs used.

The Active Ethernet cards are also interoperable with third party Active Ethernet devices.

The Active Ethernet cards supports Layer 2 bridging functions, Layer 2 security functions, Layer 3 routing functions and the Zhone Multimedia Traffic Management functionality (MTM).

- GPON

MXK-GPONX4-IO

MXK-GPONX8-IO

A quad or octal interface that supports 2.5 Gbps downstream bandwidth and 1.25 Gbps upstream bandwidth per interface as specified in the G.984.1-4 specifications.

The MXK 8 port GPON card can support up to 512 GPON subscribers using Class B+ optics. The MXK 4 port GPON card can support up to 256 GPON subscribers using Class B+ optics.

- ADSL2+

MXK-ADSL2+-BCM-48A

This card is a single slot card that supports ADSL2+ Annex A/M. The standards supported are ANSI T1.413 Issue 2, G.992.1 (G.dmt), G.992.2 (G.lite), and ADSL2+ (G.992.5) standards.

MXK-ADSL2+-POTS-BCM-48A-2S

This card is a two-slot card and provides 48 ports of integrated ADSL and POTS VoIP service. This card supports the ANSI T1.413 Issue 2, G.992.1 (G.dmt) and G.992.2 (G.lite), G.992.3 and G.992.4 (ADLS2), G.992.5 (ADSL2+), Annex A, and Annex M ADSL standards. Also supports SIP, SIP-PLAR, H.248, and MGCP protocols.

MXK-ADSL2+-SPLTR600-BCM-48A-2S

MXK-ADSL2+-SPLTR900-BCM-48A-2S

These cards are two-slot cards with an integrated POTS splitter to provide 48 ports of integrated ADSL and POTS service. Each of these lines are combined with the ADSL2+ signal internally and exits the line card in the subscriber direction with both ADSL and POTS on the loop. In the network direction the POTS is split from the ADSL signal keeping POTS on copper pairs and placing the ADSL data information on the IP network.

The MXK-ADSL2+-SPLTR600-BCM-48-2S, and MXK-ADSL2+-SPLTR900-BCM-48-2S cards support the ANSI T1.413 Issue 2, G.992.1 (G.dmt) and G.992.2 (G.lite), G.992.3 and G.992.4 (ADSL2), G.992.5 (ADSL2+), Annex A standards and Annex M ADSL standards.

The MXK-ADSL2+-BCM-48A, is a single slot card that supports ADSL2+ Annex A/M. The standards supported are ANSI T1.413 Issue 2, G.992.1 (G.dmt), G.992.2 (G.lite), and ADSL2+ (G.992.5) standards.

All ADSL cards support VoIP POTS services.

- EFM-SHDSL

The MXK SHDSL 24-port cards provides Ethernet over SHDSL links to Zhone EtherXtends and N2N CPE devices. SHDSL links can be added or removed as the network is configured. The card automatically performs load balancing over the links.

MXK-EFM-SHDSL-24-NTP

This card provides network timing reference and line power. The timing reference enables the card to use the MXK timing as the SHDSL line clocking. This allows an SHDSL CPE to derive timing from the input of the SHDSL lines. It then can use that timing/clocking to provide timing to other subtended devices. The line power feature can be used to power CPEs such as the SkyZhone to eliminate

the need for local power. The power is combined with the data and sent out over the 24 SHDSL ports to downstream CPE devices such as the SkyZhone. One MXK-EFM-SHDSL-24-NTP line card can provide power and data for up to 12 CPE devices.

MXK-EFM-SHDSL-24-NTWC

This card provides network timing reference and current. The network timing reference allows SHDSL lines to use the backplane clock to clock T1/E1 traffic eliminating the need for a clock source at each location where remote devices are installed.

– MXK-VDSL2-24-BCM

This card is a single-slot 24-port VDSL2 subscriber line card, which provides high symmetric and asymmetric bandwidth and supports 17a profile.

The MXK-VDSL2-24-BCM card can be used with the Zhone VDSL2 CPE devices. This architecture allows VDSL2 users to access the maximum bandwidth available over twisted-pair, copper phone lines.

– MXK-MTAC/RING

MXK-MTAC/RING-ENH

The MXK-MTAC/RING card is a single slot card that supports metallic loop testing for DSL and POTS interfaces with the external test set.

Note that the type of tests provided will vary, depending on the type of card being tested.

It also supports external alarm inputs (12 circuits, wet or dry, normally open or normally closed), T1/E1 or BITS external network clock access, and ring generation (internal ring generator or access for an external ring generator).

MXK-MTAC/RING-ENH card is a single slot card that is the enhance version of the MXK-MTAC/RING card. In addition to the features in the MXK-MTAC/RING card, it also supports look-out internal line test (i.e. metallic loop testing without external test set).

The MXK has a non-blocking architecture with a high-speed backplane. Each line card on the MXK had a dedicated backplane trace to each of the uplink cards.

The MXK chassis, uplink cards, line cards, and SFPs are temperature hardened.

MXK features

This section describes some key features of the MXK, including:

- [Ethernet services, page28](#)

- [GPON, page28](#)
- [VoIP, page28](#)
- [MGCP, page28](#)
- [SIP, page29](#)
- [Redundancy on uplinks, page30](#)
- [Management, page30](#)
- [Data services, page30](#)
- [Rate Limiting, page32](#)

Ethernet services

Active Ethernet cards are two-slot card or one-slot cards that support SFPs that can provide copper and fiber services and supports distances as high as 80 Km.

See [Small form factor pluggables on page492](#) for information on SFPs for Ethernet.

GPON

The 4-port and 8-port GPON cards allow per-port speeds of 2.5 Gbps downstream (on 1490 nm) and 1.25 Gbps upstream (1310nm). The GPON cards support a simple SC connector SFP with a Burst receive GPON OLT transceiver.

See [Small form factor pluggables on page492](#) for information on SFPs for GPON.

VoIP

Voice over IP, also known as Internet Telephony, supports full duplex transmission of voice traffic over IP networks. The MXK supports Media gateway control protocol (MGCP) and Session Initiation Protocol (SIP).

MGCP

Media gateway control protocol (MGCP) provides the means to interconnect a large number of IP telephony gateways. MGCP assumes that a call agent (CA) performs the intelligence of all call-control operations and that a media gateway (MG) carries out all media processing and conversion.

MGCP provides an internetworking control system to control telephony gateways from external call control elements are referred to as call agents. A telephony gateway is a network element that provides conversion between the

audio signals carried on telephone circuits and data packets carried over the Internet or over other packet networks.

MGCP assumes a call control architecture in which the call control “intelligence” is outside the gateways and handled by external call control elements. The MGCP assumes that these call control elements, or Call Agents, will synchronize with each other to send coherent commands to the gateways under their control. MGCP does not define a mechanism for synchronizing Call Agents. MGCP is, in essence, a master/slave protocol, where the gateways are expected to execute commands sent by the Call Agents.

MGCP assumes a connection model constructed of endpoints and connections. Endpoints are sources or sinks of data and could be physical or virtual.

Examples of physical endpoints are:

- An interface on a gateway that terminates a trunk connected to PSTN switch (for example, a Class 5 or Class 4 switch). A gateway that terminates trunks is called a trunk gateway.
- An interface on a gateway that terminates an analog POTS connection to a phone, key system, PBX, etc. A gateway that terminates residential POTS lines (to phones) is called a residential gateway.
- An example of a virtual endpoint is an audio source in an audio-content (media) server.

Creation of physical endpoints requires hardware installation, while creation of virtual endpoints can be done in software.

Connections may be either point-to-point or multipoint. A point-to-point connection is an association between two endpoints with the purpose of transmitting data between these endpoints. Once this association is established for both endpoints, data transfer between these endpoints can take place.

The MXK also supports Megaco, H.248.

SIP

Session Initiation Protocol (SIP) is a signaling protocol that provides a mechanism for:

- call establishment
- call teardown
- call control
- other supplementary services in an IP network.

There are two major architectural components within SIP: the SIP user agent (UA) and the SIP network server. The UA is the end system component

responsible to initiate and answer calls. The SIP server is the network device that handles the signaling associated with multiple calls.

The UA itself has a client element, the User Agent Client (UAC) and a server element, the User Agent Server (UAS). The client element initiates the calls and the server element answers the calls. This allows peer-to-peer calls to be made using a client-server protocol.

The main function of the SIP server is to provide name resolution and user location, since the caller is unlikely to know the IP address or host name of the called party, and to pass on messages to other servers or SIP endpoints. Other functions performed by the SIP servers are redirecting, forking, and registration.

Together these components make up a basic SIP infrastructure. Application servers can sit above these components delivering SIP supplementary services to end users.

Redundancy on uplinks

The MXK supports uplink card redundancy.

When the cards boot up, they elect an active and a standby card based on their respective weights. If an active card fails, the standby takes over and becomes active. Note that redundancy is non-revertive. That is, a previously active card does not become active when it starts up again.

When the standby card comes up, the active card copies over the configuration database, routing tables, and software binaries to the standby card. As configuration changes are made to the active card, the standby card is automatically updated.

Management

The MXK can be managed either in-band (VLAN tagged) on uplink Ethernet ports, out-of-band on the 10/100 Ethernet interface, or IP on a bridge.

The uplink card also contains a serial (craft) port for local management.

After establishing a connection to the MXK, administrators can manage the device using the Command Line Interface (CLI), Web UI, ZMS, or SNMP.

Data services

The MXK provides access and aggregation routing functions to connect subscribers to networks. The following MXK interfaces support IP traffic:

- One Ethernet interface on the uplink card only for management.
- High speed Ethernet interfaces on the uplink cards including two 10 GE links and eight 100/1000 Ethernet links.

The MXK provides the following key data services:

- IP forwarding and routing—incoming packets from an interface are forwarded to the appropriate output interface using the routing table rules.
- Bridging—incoming packets from an interfaces are forwarded based on MAC addresses or Layer 2 forwarding rules.
- DHCP servers and relay for IP address configuration.
- IP filtering. IP filtering is typically performed to enhance network security by limiting access between two networks.
- Numbered or unnumbered interfaces.
- Bridging: uplink, downlink, TLS, and intralinks.
- Bridging enhancements:
 - IP on a TLS bridge
 - Intralink support including multiple intralinks
 - VLAN wildcard for Q-in-Q
 - DHCP relay
- Routing (uplinks, Active Ethernet)
- Video: Multicast (IGMPv1 / v2), IGMP snooping, IGMP proxy reporting
- QoS: rate limiting (three color policing; color blind, 802.1p)
- RIP v1 (RFC 1058) RIPv2 (RFC 2453)
- DHCP server (RFC 2131, 2132)
- Broadcast storm protection
- QoS: Rate limiting, 3 color policing, 802.1p
- Link aggregation
- Q-in-Q (Active Ethernet, GPON)
- Security
 - System security: SSH, HTTPS, and SFTP
 - Secure bridging: Destination MAC swapping, secure bridging filters
- RSTP on uplinks
- GPON
 - Smart OMCI: interoperability with third party ONTs
 - 64 splits, class B+ optics
 - Dynamic GEM port creation

The MXK can be managed with:

- Command line interface (CLI)
- ZMS

- WebUI

Rate Limiting

Rate limiting is a mechanism for controlling traffic and can include policing (dropping packets). Use rate limiting to control the rate of traffic sent or received on the ingress or the egress of both the logical port or the physical port of the MXK. Traffic that is less than or equal to the specified rate is sent and traffic that exceeds the rate is dropped. The rate limiting does not include queuing which delays packets in a buffer.

After configuring an interface with rate limiting, the traffic rate is monitored and metered to verify conformity with an established contract.

Non-conforming traffic is discarded, while conforming traffic passes through the interface without any changes. The MXK follows RFC 2697 for rate limiting on both the ingress and egress of the interface.

2 MXK OPERATIONS, ADMINISTRATION, AND MAINTENANCE

This chapter describes MXK operations, system administration, and maintenance functions:

- [MXK management interfaces, page33](#)
- [MXK card configuration, page49](#)
- [MXK system administration, page53](#)
- [Map subscriber information to a port description field on the MXK, page82](#)
- [MXK security, page87](#)
- [MXK clocking, page98](#)
- [MXK alarms, page100](#)

MXK management interfaces

This section describes how to access and navigate the MXK, configure management interfaces, and manage the MXK with the Web UI:

- [Log in to the serial \(craft\) port, page33](#)
- [MXK default configuration, page35](#)
- [Configure management interfaces, page37](#)
- [Configure DNS resolver, page41](#)
- [Manage the MXK with the Web User Interface, page43](#)
- [Disable partial config sync traps to ZMS from the device, page46](#)

Log in to the serial (craft) port

This section describes how to configure the serial craft port and log in and out of the MXK:

- [Configure the MXK serial craft settings, page34](#)
- [Change the serial craft port settings, page34](#)

- [Log in and out of the system, page35](#)

Configure the MXK serial craft settings

Since the MXK has a passive chassis, you must install the uplink card in slot *a* before you can log in to the serial port and begin the initial configuration of the system.

The MXK unit provides an out-of-band RS232 D serial (craft) interface for managing the unit. To access the serial port on the uplink card, configure the **rs232-profile** with these settings:

- 9600bps
- 8 data bits
- No parity
- 1 stop bit
- No flow control



Note: Do not use the serial craft port of a standby card to modify its configuration.

Tip: The serial (craft) port settings can be changed by modifying the **rs232-profile**.

You must perform the initial configuration of the system using the serial (craft) interface. After completing the initial configuration, you can manage the MXK unit over the network through a telnet session over the Ethernet interface.



Note: The MXK supports six concurrent management sessions, five telnet sessions and a single local session through the serial (craft) port.

Change the serial craft port settings

Update the **rs232-profile** for the shelf and slot that contain the serial craft port.

Caution: The serial craft port supports speeds of 9600, 19200, 38400, and 57600. Do not set the speed to an unsupported value. Doing so could render the serial craft port inaccessible.

To update the **rs232-profile** enter:

```
zSH> update rs232-profile 1-a-1-0/rs232
rs232-profile 1-a-1-0/rs232
Please provide the following: [q]uit.
rs232PortInSpeed: -----> {9600}: 57600
rs232PortOutSpeed: -----> {9600}: 57600
```

```
rs232PortInFlowType: ----> {none}:
rs232PortOutFlowType: ---> {none}:
rs232AsyncPortBits: -----> {8}:
rs232AsyncPortStopBits: -> {one}:
rs232AsyncPortParity: ---> {none}:
rs232AsyncPortAutobaud: -> {disabled}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Log in and out of the system

When you first access the MXK, the default login is **admin** and the default password is **zhone**:

```
login: admin
password:
zSH>
```

To log out of the system, enter the **logout** command:

```
zSH> logout
```

The system automatically logs you out after a period of inactivity. The default logout time is 10 minutes. To change the logout time enter the time-out command with the time in minutes:

```
zSH> timeout 120
CLI time-out value is now at 120 minutes.
```

To reset time-out to the default enter:

```
zSH> timeout -d
CLI time-out value reset to default of 10 minutes.
```

MXK default configuration

This section describes the MXK system defaults, monitoring the MXK, and temporary logging sessions:

- [Defaults overview, page35](#)
- [Monitoring the MXK through the serial craft port, page36](#)
- [Enable/disable temporary logging sessions, page36](#)

Defaults overview

The MXK must have at least one uplink card installed before the MXK will boot properly. Along with the ability to display cards (both active and inactive) which are in the MXK, you can also see into the DOS file system which stores boot code, software images, and configurations. See [Navigate](#)

[the MXK file system on page 65](#) for a description of commands which can be used to access the MXK file system.

Line cards (except the first uplink card in slot *a*) must be provisioned with a **card-profile** before they will boot up.

- Administrative user name is **admin**, password is **zhone**.
- A single record for the Ethernet interface on the uplink card in slot *a* exists. No other profiles to configure physical interfaces exist.
- The uplink card in slot *a* is enabled. You must enable all other cards including the uplink card in slot *b* in a **card-profile** before they will boot up.
- A default **system 0** profile exists with the following configuration:
 - Authentication traps are not enabled
 - ZMS communication is not configured
 - Alarm notification and output are enabled for all severity levels

Monitoring the MXK through the serial craft port

The MXK can send messages to a console session, a log file, or to a syslog server and be configured to a number of system event levels — emergency, alert, critical, error, warning, notice, information, and debug.

By default logging is enabled on the serial craft port and disabled over telnet sessions. To enable or disable logging for the session, using the following command:

Enable/disable temporary logging sessions

By default, log messages are enabled on the serial craft port. Use the **log session** command and the **log serial** command to enable/disable logging:

The **log session** command enables/disables logging messages for that session only when connected to the device through a Telnet session. If the user logs out, the logging setting returns to the default. To enable/disable logging for the current Telnet session only: enter

```
zSH> log session on
zSH> log session off
```

The **log serial** command enables/disables logging messages for the session on the serial craft port. This command can be used in both Telnet connections and serial port connections to turn on and off the serial craft port logs. To enable/disable logging for the serial craft port enter:

```
zSH> log serial on
zSH> log serial off
```

This command setting persists across system reboots.

Configure management interfaces

This section describes the three ways to configure the MXK for management:

- [Configure an out-of-band IP management interface, page37](#)
- [Configure an in-band management interface, page38](#)
- [Configure IP on a bridge for MXK management, page38](#)
- [Create the default route, page41](#)

Configure an out-of-band IP management interface

The MXK has a 10/100 BaseT Ethernet interface on the uplink card for out-of-band management. The **ip-interface-record** profile for this interface is **ethernet1**. This interface is shared between the two Ethernet ports on redundant uplink cards (if they exist). The system can be reached using the address configured in the **ethernet1 ip-interface-record**, no matter which card is active.

Caution: You must configure the Ethernet interface on the uplink card before any other interfaces on the system, even if you do not intend to manage the unit over the Ethernet.

Configuring an out-of-band IP management interface

The following example configures the IP address for out-of-band management of the MXK. Refer to the *CLI Reference Guide* for a complete description of the command options and syntax.

- 1 Configure the 10/100 Ethernet interface on the uplink card.

```
zSH> interface add1-a-1-0/eth 192.168.8.21/24
Created ip-interface-record ethernet1/ip.
```

- 2 Verify the interface.

```
zSH> interface show
1 interface
Interface      Status  Rd/Address          Media/Dest Address  IfName
-----
1/a/1/0/ip     UP      1 192.168.8.21/24   00:01:47:17:ee:54   ethernet1
-----
```

- 3 Create the default route.

See [Creating a default route on page 41](#).

Configure an in-band management interface

Configuring an in-band management interface

The following example configures the IP address for in-band management of the MXK.

- 1 Configure an uplink Ethernet port for in-band management.

```
zSH> interface add 1-a-2-0/eth vlan 99 192.168.8.21/24
Created ip-interface-record ethernet2-99/ip.
```

Verify the interface.

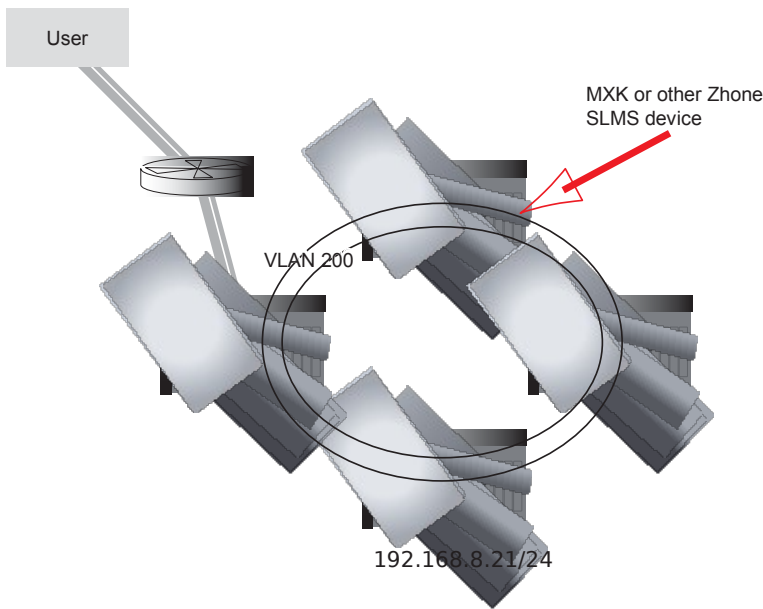
```
zSH> interface show
1 interfaces
Interface      Status  Rd/Address          Media/Dest Address  IfName
-----
1/a/2/0/ip     UP      1 192.168.8.21/24   00:01:47:17:ee:55   ethernet2-99
```

- 2 Create the default route.
See [Creating a default route on page 41](#).

Configure IP on a bridge for MXK management

IP on a bridge allows you to put an IP address on a bridged VLAN for managing the MXK. This VLAN can be used to manage multiple MXKs or other devices. One IP on a bridge can be created per MXK.

Figure 3: IP on a bridge



Before configuring IP on a bridge, configure one bridge of the type you wish to use for your IP on a bridge configuration. Otherwise the following error message appears:

```
zSH> interface add 1-a-6-0/ipobridge vlan 100 192.168.123.2/24
Error: Couldn't determine type of IPOBRIDGE to create!
      Create an 'uplink' or 'tls' bridge(s) first.
```

Create IP on a bridge for TLS



Note: The logical port interface for IP on a bridge on the MXK *must* be *1-a-6-0/ipobridge* for correct transmission of IP packets.

Creating IP on a bridge interfaces (TLS)

Create an IP on a bridge interface using the IP address of 192.168.8.21/24, and a logical port interface 6 with a VLAN 200.

- 1 Enter **interface add** *interface/type* with the type as *ipobridge*:

```
zSH> interface add 1-a-6-0/ipobridge vlan 200 192.168.8.21/24
Created ip-interface-record ipobridge-200/ip.
```

This command creates the new IP interface as well as a new bridge. The bridge created will be a Transparent Lan Service (TLS) tagged bridge.

- 2 Verify the *ipobridge* interface:

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address          Media/Dest Address  IfName
-----
1/a/1/0/ip     UP      1 172.16.160.49/24  00:01:47:17:ee:54   1-a-1-0
1/a/6/0/ip     UP      1 192.168.8.21/24   00:01:47:17:ee:54   ipobridge-200
-----
```

- 3 Verify the TLS bridge:

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
upl ST  102/502  ethernet5-102-502/bridge           DWN
tls Tagged 200  ipobridge-200/bridge              UP   D 00:01:47:17:ee:54
```

- 4 Create the upstream bridge with the same VLAN for the management interface, then verify the bridge created with **bridge show**: This bridge *must* be a TLS bridge and use the same VLAN ID.

```
zSH> bridge add 1-a-4-0/eth tls vlan 200 tagged
Adding bridge on 1-a-4-0/eth
Created bridge-interface-record ethernet4-200/bridge
```

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
```

```

-----
upl ST 102/502 ethernet5-102-502/bridge DWN
tls Tagged 200 ipobridge-200/bridge UP D 00:01:47:17:ee:54
tls Tagged 200 ethernet4-200/bridge UP

```

The uplink card is now reachable from the upstream, and IP 192.168.8.21/24 can reach other upstream devices on the same VLAN.

Follow the same steps to create an IP on a bridge and bridges for downstream devices.

The IP on a bridge feature does not support SNMP.

5 Create the default route.

See [Creating a default route on page 41](#).

Create IP on a bridge on an uplink bridge

To create IP on a bridge on an uplink bridge, you must create the uplink bridge and the bridge-path before creating the *ipobridge* interface. The *ipobridge* interface reads the bridge-path profile in order to determine the type of *ipobridge* to create.

Creating IP on a bridge on a uplink bridge

1 Create a bridge designating *uplink* and a VLAN:

```

zSH> bridge add 1-a-2-0/eth uplink vlan 400
Adding bridge on 1-a-2-0/eth
Created bridge-interface-record ethernet2-400/bridge

```

2 Create the bridge-path for the uplink bridge:

```

zSH> bridge-path add ethernet2-400/bridge vlan 400 default
Bridge-path added successfully

```



Note: All bridge paths are set to *default* on the MXK.

3 Create the *ipobridge* interface using the same VLAN:

```

zSH> interface add 1-a-6-0/ipobridge vlan 400 192.168.8.21/24
Created ip-interface-record ipobridge-400/ip.

```

4 Verify the *ipobridge* interface:

```

zSH> interface show
2 interfaces
Interface      Status  Rd/Address      Media/Dest Address IfName
-----
1/a/1/0/ip     UP      1 172.16.160.49/24  00:01:47:1a:fe:64 ethernet1
1/a/6/0/ip     UP      1 192.168.8.21/24   00:01:47:1a:fe:64 ipobridge-400
-----

```


5 Verify the ipobridge and the uplink bridge:

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
upl Tagged 400  ethernet2-400/bridge                DWN S VLAN 400 default
dwn Tagged 400  ipobridge-400/bridge                UP  D 00:01:47:1a:fe:64
```

IP on a bridge automatically creates a downlink bridge with the same VLAN ID.

6 Create the default route.

See [Creating a default route on page 41](#).

Create the default route

Regardless of which management IP interface is created, you must also create a default route for that interface.

Creating a default route

Create the default route using the gateway 192.168.8.1 with a cost of 1 (one).

```
zSH> route add default 192.168.8.1 1
```

Verify the route:

```
zSH> route show
Destination Routing Table
Dest                               Nexthop                Cost    Owner      Fallback
-----
0.0.0.0/0                          192.168.8.1            1       STATICLOW
192.168.8.0/24                     1/a/6/0/ip             1       LOCAL
```

Use the **ping** command to verify connectivity to the default gateway:

```
zSH> ping 192.168.8.1
PING 192.168.8.1: 64 data bytes
!!!!
----192.168.8.1 PING Statistics----
5 packets transmitted, 5 packets received
round-trip (ms)  min/avg/max = 0/0/0
```

To stop the ping, press CTRL+C.

Configure DNS resolver

Domain Name System (DNS) maps domain names to IP addresses, enabling the system to reach destinations when it knows only the domain name of the destination. In other words, you can use **ping** and a name instead of an IP address. DNS configuration uses the following profiles:

- **resolver**—Configures the global DNS resolver, including the DNS search order, default domain name, and list of nameserver addresses. The DNS settings in this record can be used for local applications by administrators on the system, such as **traceroute** or **ping**.
- **host-name**—A replacement for the UNIX local hosts table. Up to four host aliases can be defined for each host entry. Settings in the **resolver** record determine whether the hosts table is searched.

[Table 3](#) describes the configurable parameters for the resolver profile (all others should be left at their default values):

Table 3: Configurable resolver parameters

Parameter	Description
query-order	The kind of resolver query for this routing domain. Values: hosts-first searches the local hosts table first then the list of nameservers. dns-first searches the list of nameservers first then the local hosts table. dns-only searches only the list of nameservers. Default: hosts-first
domain	The routing domain to which this host parameter applies. The default is an empty string. The only routing domain supported is domain 1.
first-nameserver	The IP address of the first or primary nameserver for this routing domain. The default value is 0.0.0.0.
second-nameserver	The IP address of the second or secondary nameserver for this routing domain. This nameserver is queried if the first nameserver cannot resolve the query. The default value is 0.0.0.0.
third-nameserver	The IP address of the third or tertiary nameserver for this routing domain. This nameserver is queried if the first nameserver cannot resolve the query. The default value is 0.0.0.0.

The following example creates a **resolver** record for a routing domain:

```
zSH> new resolver 1
Please provide the following: [q]uit.
query-order: -----> {hosts-first}:
domain: -----> {}: zhone.com
first-nameserver: --> {0.0.0.0}: 192.168.8.21
second-nameserver: -> {0.0.0.0}: 201.23.20.2
third-nameserver: --> {0.0.0.0}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.
```

Another way to create DNS is by creating a **hosts** profile after the **resolver** profile is created. The syntax is **new host-name routingdomain/ipoctet1/ipoctet2/ipoctet3/ipoctet4**.

[Table4](#) describes the configurable parameters in the **host-name** profile (all others should be left at their default values).

Table 4: Configurable parameters in the host-name profile

Parameter	Description
hostname	Client host name (if any) that the client used to acquire its address. The default is an empty string.
hostalias1	Host name alias for the specified host. The default value is an empty string.
hostalias2	Secondary host name alias for the specified host. The default value is an empty string.
hostalias3	Tertiary host name alias for the specified host. The default value is an empty string.
hostalias4	Quaternary host name alias for the specified host. The default value is an empty string.

```

zSH> new host-name 1/192/168/8/32
Please provide the following: [q]uit.
hostname: ---> {}: www.zhone.com
ipaddress: --> {0.0.0.0}: 192.168.8.32
hostalias1: -> {}: engineering.zhone.com
hostalias2: -> {}: marketing.zhone.com
hostalias3: -> {}: sales.zhone.com
hostalias4: -> {}: gss.zhone.com
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.

```

Manage the MXK with the Web User Interface

This section describes:

- [Manage the MXK using Zhone Web User Interface, page44](#)
- [Disable the Web UI, page45](#)

Before using Zhone Management System (ZMS), the Web UI or any remote management, the management interface must configured. See [Configure management interfaces on page37](#).

Caution: If you are using a public and not a private IP address for the Web UI, to protect your management system, Zhone recommends that the port access profile is configured for the Telnet port (port 23) and the management subnet is specified. See [Port access security on page91](#) for more information on setting up port security.

The MXK enables Web-based configuration using the Zhone SLMS Web Interface Tool. The Zhone SLMS Web Interface Tool supports configuration and management of both line and uplink cards.

Manage the MXK using Zhone Web User Interface

To manage the MXK using the Zhone Web User Interface (UI):

- Add an IP address to the interface to be used for management.

On the uplink cards, the interface on the 10/100 Ethernet port or GigE ports can be used. Ensure that the IP address is in the same subnet as the client devices and is reachable through Telnet. This example adds an IP interface for 172.24.94.103 to the 10/100 Ethernet port using VLAN 94.

```
zSH> interface add 1-1-1-0/eth vlan 94 172.24.94.103/24
Created ip-interface-record ethernet1-94/ip
```

- Configure a default route to the IP interface.

The default route enables connectivity to the IP interface.

```
zSH> route add default 94 172.24.94.103 metric 1
```

To launch the Zhone Web User Interface, in a browser URL address space on a PC with connectivity to the MALC, enter the IP address configured on the MXK.

The Zhone Web User Interface launches and displays the Login window for the MXK.

Figure 4: Zhone Web User Interface Login Screen



MXK823

The Multi-Access Line Concentrator, MXK, platform provides low-cost, high-density subscriber access concentration in the Zhone Single Line Multi-Service (SLMS) architecture. The MXK is designed as a high-bandwidth, non-blocking Ethernet based access platform that integrates high bandwidth access technologies (GPON and active Ethernet) of core/metro Ethernet networks with the breadth and flexibility of interfaces required at the edge of networks.

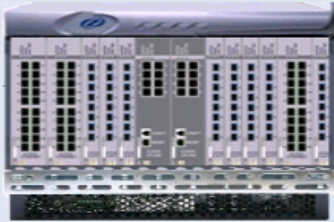
The MXK can be deployed in Central Office environments or controlled environmental vaults for remote terminal applications. The MXK is intended for restricted access locations only.

Web Interface Login

User Name

Password

MXK 823



On the Login page, enter the user name and password. The default user name is *admin* and the default password is *zhone*.



Note: Zhone recommends you change the user name and password to ones suitable to your network.

Click the desired menu to display the management options. For online help, click the Help icon  or product title in any window.

Disable the Web UI

Disabling the Web UI

Delete the *mxk823_http.tar* or *mxk819_http.tar* file from the *card1* directory to remove the software file associated with the Web UI. If you remove this file, you would have to reinstall the file in the *card1* directory to run the Web UI.

- 1 Verify the current directory.

```
zSH> pwd
/card1
```

- 2 Verify the *mxk823_http.tar* or *mxk819_http.tar* file in the *card1* directory.

```
zSH> dir
Listing Directory ..
drwxrwxrwx  1 0      0          4096 Apr 20 09:18 datastor/
-rwxrwxrwx  1 0      0          701448 Apr 20 09:12 mxup2tg8graw.bin
-rwxrwxrwx  1 0      0      10653900 Apr 20 09:12 mxup2tg8g.bin
-rwxrwxrwx  1 0      0      5789248 Mar 19 07:18 mxlc4gp.bin
-rwxrwxrwx  1 0      0      5741688 Apr 20 09:12 mxlc8gp.bin
drwxrwxrwx  1 0      0          4096 Mar 11 07:24 onreboot/
drwxrwxrwx  1 0      0          4096 Nov 25 2008 pub/
drwxrwxrwx  1 0      0          4096 Apr 20 09:15 log/
drwxrwxrwx  1 0      0          4096 Dec 31 1999 bulkstats/
-rwxrwxrwx  1 0      0      8895488 Apr 20 09:11 mxk823_http.tar
```

- 3 Delete the *mxk823_http.tar* file.

```
zSH> del mxk823_http.tar
```

The file is removed from the MXK. The file must be reinstalled in the *card1* directory to run the Web UI.

Disable partial config sync traps to ZMS from the device

Before using Zhone Management System (ZMS), the Web UI or any remote management, the management interface must be configured. See [Configure management interfaces on page37](#).



Note: For how to enable ZMS, refer to the *NetHorizon User's Guide*.

CLI mass provisioning and ZMS

If you need to perform mass provisioning tasks with a script from the CLI when ZMS is managing the device, you must first disable ZMS in the **system 0** profile, complete the mass provisioning, enable ZMS again, and perform a config sync in ZMS.

- 1 Disable ZMS from managing the device, change the **zmsexists** parameter from *true* to *false*:

```
zSH> update system 0
system 0
Please provide the following: [q]uit.
syscontact: -----> {Zhone Global Services and Support 7001 Oakport
Street Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113
support@zhone.com}:
```

```

sysname: -----> {Zhong MxK}:
syslocation: -----> {Oakland}:
enableauthtraps: -----> {disabled}:
setserialno: -----> {0}:
zmsexists: -----> {true}: false
zmsconnectionstatus: --> {inactive}:
zmsipaddress: -----> {0.0.0.0}:
configsyncexists: -----> {false}:
configsyncoverflow: ---> {false}:
configsyncpriority: ---> {high}:
configsyncaction: -----> {noaction}:
configsyncfilename: ---> {}:
configsyncstatus: -----> {syncinitializing}:
configsyncuser: -----> {}:
configsyncpasswd: -----> {** private **}: ** read-only **
numshelves: -----> {1}:
shelvesarray: -----> {}:
numcards: -----> {3}:
ipaddress: -----> {0.0.0.0}:
alternateipaddress: ---> {0.0.0.0}:
countryregion: -----> {us}:
primaryclocksource: ---> {0/0/0/0/0}:
ringsource: -----> {internalringsource}:
revertiveclocksource: -> {true}:
voicebandwidthcheck: --> {false}:
alarm-levels-enabled: -> {critical+major+minor+warning}:
userauthmode: -----> {local}:
radiusauthindex: -----> {0}:
secure: -----> {disabled}:
webinterface: -----> {enabled}:
options: -----> {NONE(0)}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

- 2 Enable ZMS to manage the device, change the **zmsexists** parameter from *false* to *true*:

```

zSH> update system 0
system 0
Please provide the following: [q]uit.
syscontact: -----> {Zhong Global Services and Support 7001 Oakport
Street Oakland Ca. (877) Zhong20 (946-6320) Fax (510)777-7113
support@zhong.com}:
sysname: -----> {Zhong MxK}:
syslocation: -----> {Oakland}:
enableauthtraps: -----> {disabled}:
setserialno: -----> {0}:
zmsexists: -----> {true}: true
zmsconnectionstatus: --> {inactive}:
zmsipaddress: -----> {0.0.0.0}:
configsyncexists: -----> {false}:
configsyncoverflow: ---> {false}:
configsyncpriority: ---> {high}:

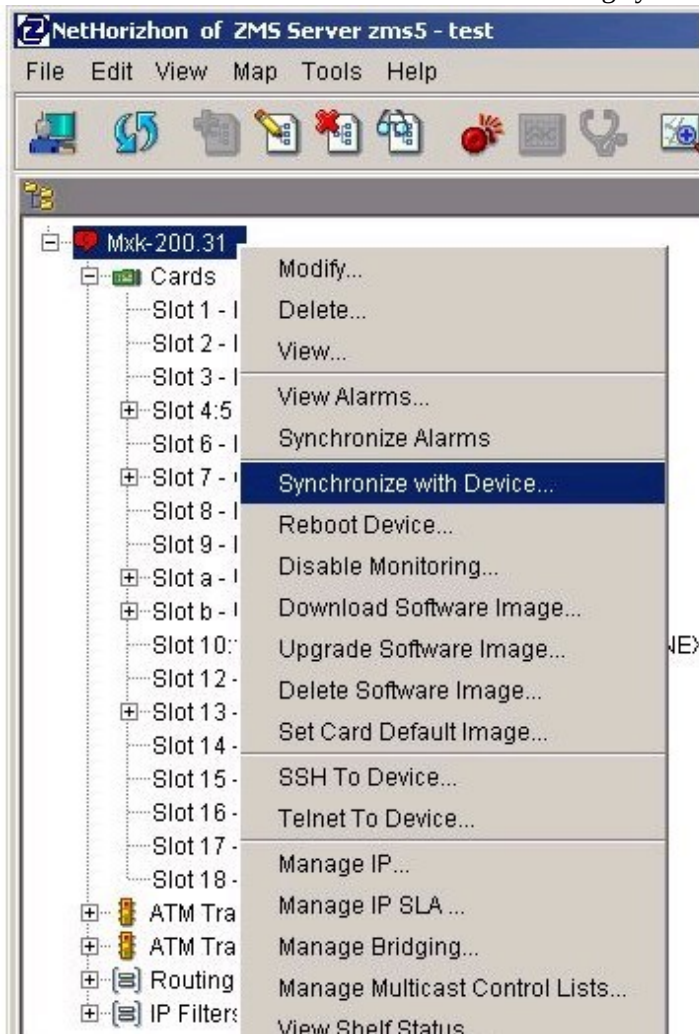
```

```

configsyncaction: -----> {noaction}:
configsyncfilename: ---> {}:
configsyncstatus: -----> {syncinitializing}:
configsyncuser: -----> {}:
configsyncpasswd: -----> {** private **}: ** read-only **
numshelves: -----> {1}:
shelvesarray: -----> {}:
numcards: -----> {3}:
ipaddress: -----> {0.0.0.0}:
alternateipaddress: ---> {0.0.0.0}:
countryregion: -----> {us}:
primaryclocksource: ---> {0/0/0/0/0}:
ringsource: -----> {internalringsource}:
revertiveclocksource: -> {true}:
voicebandwidthcheck: --> {false}:
alarm-levels-enabled: -> {critical+major+minor+warning}:
userauthmode: -----> {local}:
radiusauthindex: -----> {0}:
secure: -----> {disabled}:
webinterface: -----> {enabled}:
options: -----> {NONE(0)}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```


3 Perform a full config sync in ZMS.



Note: For details on using ZMS, refer to the *ZMS Administrator's Guide* and the *NetHorizon User's Guide*.

CLI configuration of a device being managed by the ZMS is disabled by default. Attempting to configure the device results in an error:

MXK card configuration

This section describes how to provision MXK cards:

- [View uplink cards, page50](#)
- [View line cards, page50](#)
- [MXK card configuration on page51](#)

View uplink cards

You can view information by entering the **slots** command with the uplink card slot of the uplink card including:

- ROM Version
- Software Version
- Card-Profile ID

The asterisk next to the type of card indicates that this card is in a redundant configuration.

```
zSH> slots a
Type           : *MXK TWO TENGIGE EIGHT GIGE
Card Version   : 800-02485-01-A
EEPROM Version : 1
Serial #       : 1769060
CLEI Code      : No CLEI
Card-Profile ID : 1/a/10100
Shelf          : 1
Slot           : a
ROM Version    : MALC CAN MXK-20-Feb-2007
Software Version: MXK 1.16.1.128
State          : RUNNING
Mode           : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat  : 52
Fault reset    : enabled
Uptime         : 18 days, 1 hour, 31 minutes
```

View line cards

After you install the uplink card in slot *a*, all other line cards and the uplink card in slot *b* (for redundant configurations) must be provisioned.

The **slots** command shows the cards currently exist in the MXK chassis and their state including: running, loading, not provisioned, booting, and configuring.

```
zSH> slots
Uplinks
a: *MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
4: MXK 4 PORT GPON (RUNNING)
13: MXK 20 ACT ETH (RUNNING)
```

Entering the slots command with the slot number displays information on that particular card. In this case, entering **slots a** displays information about the line card in slot 13. You can find the ROM, software version, and other card information.

```

zSH> slots 13
Type           : MXK 20 ACT ETH
Card Version   : 800-02316-01-A
EEPROM Version : 1
Serial #       : 1769100
CLEI Code      : No CLEI
Card-Profile ID : 1/13/10200
Shelf          : 1
Slot           : 13
ROM Version    : MXK 1.16.0.128
Software Version: MXK 1.16.1.128
State          : RUNNING
Mode           : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat  : 52
Fault reset    : enabled
Uptime         : 18 days, 1 hour, 33 minutes

```

MXK card configuration

This section describes how to:

- [Add a card profile, page51](#)
- [Delete a card profile, page52](#)

Add a card profile

The MXK distinguishes the differences between cards and their functionality by designating a card type with the **card add** command.

To provision the cards in a MXK chassis enter **card add slotnumber**. This command automatically creates the **card-profile** for the card. The slot number determines the card type.

Adding a card profile

If necessary, use the **slots** command to verify which slot a card resides in before using the **card add** command to provision the card. To provision a card, first install the card in a slot.

- 1 To verify the location of a card, enter **slots**:

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (NOT_PROV)
Cards
  9: MXK 4 PORT GPON (NOT_PROV)
  13: MXK 20 ACT ETH (NOT_PROV)

```

- 2 To provision a card, enter **card add slotnumber**:

```
zSH> card add 13
new card-profile 1/13/10200 added, sw-file-name "mxlc20ae.bin"
```

3 To verify the state of the provisioning, enter **slots** again:

```
zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (NOT_PROV)
Cards
  9: MXK 4 PORT GPON (NOT_PROV)
  13: MXK 20 ACT ETH (LOADING)
```

When the card changes from loading to running the system sends a message:

```
zSH> JUN 29 16:23:29: notice : 1/a/12 : shelfctrl: Card in slot 13 changed
state to RUNNING.
```

After a bit, verify the state of the card again.

```
zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (NOT_PROV)
Cards
  9: MXK 4 PORT GPON (NOT_PROV)
  13: MXK 20 ACT ETH (RUNNING)
```

Delete a card profile

Deleting a card, deletes the **card-profile** interface and all provisioning including any associated routing **ip-interface-record** profiles and bridging **bridge-interface-record** profiles.

Deleting a card profile

Caution: Before deleting card profiles, perform the following:

- Back up the MXK configuration. See the release notes for information.
 - For voice cards, ensure all subscribers and voice profiles are deleted before deleting the card.
 - Remove the card from the system as explained in the *MXK Hardware Installation Guide*.
-

Delete the **card-profile** for a card to delete all the profiles associated with a card. After deleting the card, the specified card reboots.

The **card delete** command uses the following syntax:

card delete shelf/slot/cardtype

```

zSH> card delete 1/13/10200
card-profile 1/13/10200 deleted
zSH> JUN 29 16:15:35: critical: 1/13/1035: rebootserver:
* * * * Slot Reboot : type = 2, shelf = 1, slot = 13
JUN 29 16:15:34: info    : 1/a/1054: carddeletehdlr: Starting residual
profile deletions for card 1/13/10200
JUN 29 16:16:09: info    : 1/a/1054: carddeletehdlr: Residual profile
deletions in progress for card 1/13 (100 records removed)
JUN 29 16:16:10: info    : 1/a/1054: carddeletehdlr: Completed residual
profile deletions for card 1/13/10200 (113 records removed)

```

The following slots commands show the change of status of the Active Ethernet card in slot 1 immediately after entering **card delete**. The state of the card changes from running to not provisioned.

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (NOT_PROV)
Cards
  9: MXK 4 PORT GPON (NOT_PROV)
 13: MXK 20 ACT ETH (RUNNING)

```

The system also displays a message that all provisioning associated with the card is being deleted.

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  4: MXK 4 PORT GPON (RUNNING)
 13: MXK 20 ACT ETH (NOT_PROV)

```



Note: You can only delete one card at a time. Wildcards are not supported when deleting cards and their profiles.

MXK system administration

This section describes how to work with profiles in the MXK system, and many of the CLI commands that are useful when performing system administration tasks and includes a discussion of:

- [User account administration, page54](#)
- [View chassis and system information, page56](#)
- [MXK basic system administration commands, page57](#)
- [Navigate the MXK file system, page65](#)
- [Backup system configurations, page67](#)
- [SNTP, page68](#)

- [MXK Simple Network Management Protocol \(SNMP\), page69](#)
- [Use log files to monitor the system, page71](#)

User account administration

MXK users have access to the CLI and are able to configure and administer the system. This section describes the following:

- [Add users, page54](#)
- [Change default user passwords, page55](#)
- [Delete users, page55](#)
- [Delete the admin user account, page55](#)
- [Reset passwords, page56](#)

Add users

Every administrative user on the system must have a user account. The account specifies their username and password, as well as their privilege level, which determines their access to commands.

Users with **admin** privileges have access to all the administrative commands. Users with **user** privileges have access to a very limited set of commands.



Note: When entering access level responses, enter **yes** completely or the CLI interprets the response as **no**.

To add a user, enter the following commands:

```
zSH> adduser
Please provide the following: [q]uit.
User Name: jjsmith
User Prompt[zSH>]:

Please select user access levels.
admin: -----> {no}: yes
zhonedebug: --> {no}:
voice: -----> {no}:
data: -----> {no}:
manuf: -----> {no}:
database: ----> {no}:
systems: ----> {no}:
tool: -----> {no}:
useradmin: ---> {no}: yes
.....
User name:(jjsmith) User prompt:(zSH>)
Access Levels:
(admin)(useradmin)
Save new account? [s]ave, [c]hange or [q]uit: s
User record saved.
```

TEMPORARY PASSWORD: hmj4mxFU

Immediately after activating the user account, change the password. See [Change default user passwords on page 55](#).

Change default user passwords

When adding users, the system automatically assigns a temporary password to each user. Most users will want to change their password. The **changepass** command changes the password for the current logged in user. The following is an example of changing a password:

```
zSH> changepass
Current Password:
New Password:
Confirm New Password:
Password change successful.
```

Delete users

To delete a user, enter the **deleteuser** command and specify the username:

```
zSH> deleteuser jjsmith
OK to delete this account? [yes] or [no]: yes
User record deleted.
```

Delete the admin user account

In addition to deleting regular user accounts, you can also delete the **admin** user account. This account is automatically created by the system and provides full access to the CLI.



Note: You cannot delete the **admin** account (or any other user account with **useradmin** privileges) if you are currently logged into it.

To delete the **admin** account:

```
zSH> deleteuser admin
```

If desired, you can recreate an account named **admin** after deleting it:

```
zSH> adduser admin
Please provide the following: [q]uit.
User Name: admin
User Prompt[zSH>]:

Please select user access levels.
admin: -----> {no}: yes
zhonedebug: --> {no}:
voice: -----> {no}: yes
data: -----> {no}: yes
manuf: -----> {no}: yes
```

```

database: ----> {no}: yes
systems: -----> {no}: yes
tool: -----> {no}: yes
useradmin: ---> {no}: yes
.....
User name:(admin) User prompt:(zSH>)
Access Levels:
(admin)(voice)(data)(manuf)(database)(systems)(tools)(useradmin)
Save new account? [s]ave, [c]hange or [q]uit: s
User record saved.
TEMPORARY PASSWORD: hmj4mxFU

```

Reset passwords

If a user forgets their password, an administrative user can reset the password and generate a new one using the **resetpass** command, as in the following example:

```

zSH> resetpass jjsmith
Password:

```

View chassis and system information

To view overall status of the system, use the **shelfctrl monitor** command:

To view general system statistics:

```

zSH> shelfctrl monitor
Shelf uptime: 7 days, 1 hour, 46 minutes
Shelf Monitor CPLD version: 0.5
Shelf Monitor Firmware version: 0.5
Outlet temperature sensor: 22 degrees C (normal)
Fan Power A: normal
Fan Power B: normal
Power Supply A: normal
Power Supply B: normal
Fan status: OK.
System: Critical alarm set.
Card a: Critical alarm set.

```

To verify whether the shelf is active:

```

zSH> shelfctrl show
Shelf Controller Address: 01:a:12
Shelf Registry Address: 01:a:1044
Lease ID: 0x02070000_0000003a
State: active
Slot 1:
  prevState: CONFIGURING currentState: RUNNING
  mode: FUNCTIONAL startTime: 958960751

```

To view system statistics enter:

```

zSH> shelfctrl stats

```


Shelf Controller Message Statistics

```

-----
Card updates: 19
Card ECHO: 0
Directory services messages: 2
Clock messages: 2013336
Lease messages: 7736
Heartbeat messages: 2861848
Card update errors: 0
Card ECHO errors: 0
Directory services errors: 0
Clock errors: 0
Lease errors: 0
Heartbeat errors: 0
Receive errors: 0
No Peer heartbeat check errors: 0

```

MXK basic system administration commands

Commands: new, list, show, get, update, delete

This section describes these commands:

- [new command, page57](#)
- [list command, page57](#)
- [show command, page59](#)
- [get command, page61](#)
- [update command, page61](#)
- [delete command, page62](#)

new command

The **new** command can create new GPON traffic profiles.

```

zSH> new gpon-traffic-profile 1
gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}:
traffic-class: -----> {ubr}:
compensated: -----> {false}:
shared: -----> {false}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

```

list command

The **list** command displays all the profiles available on the MXK (partial list shown):

```

zSH> list
adsl-co-profile: shelf/slot/port
adsl-cpe-profile: shelf/slot/port
adsl-profile: shelf/slot/port
alarm-config: ifIndex
analog-fxo-cfg-profile: ifIndex
analog-fxs-cfg-profile: ifIndex
analog-if-cfg-profile: ifIndex
atm-cc: atmVcCrossConnectIndex
atm-if: ifIndex
atm-if-stats: ifIndex
atm-traf-descr: index
atm-traf-descr-stats: index
atm-vcl: ifIndex/vpi/vci
atm-vcl-param: index
atm-vcl-stats: ifIndex/vpi/vci
atm-vpi: ifIndex/vpi
atm-vpl: ifIndex/vpi
bridge-interface-record: ifIndex
bulk-statistic: index

```

The **list gpon-traffic-profile** command lists all GPON traffic profiles on the system.

```

zSH> list gpon-traffic-profile
gpon-traffic-profile 1
gpon-traffic-profile 2
gpon-traffic-profile 3
3 entries found.

```

The **list system** command displays the list of system profiles.

```

zSH> list system
system 0
1 entry found.

```

To view the card profiles existing on the system, enter **list card-profile**:

```

zSH> list card-profile
card-profile 1/a/10100
card-profile 1/6/10201
card-profile 1/1/10200
3 entries found.

```

To view the **bridge-interface-record** profiles of existing bridges enter **list bridge-interface-record**:

```

zSH> list bridge-interface-record
bridge-interface-record ethernet2-94/bridge
bridge-interface-record 1-1-1-0-eth-94/bridge
bridge-interface-record ethernet2-220/bridge
bridge-interface-record 1-1-1-0-eth-220/bridge
bridge-interface-record ethernet2-998/bridge
bridge-interface-record 1-1-1-0-eth-998/bridge

```

6 entries found.

show command

Use the **show** command to view all the options in a profile. For example, if you need to find which country codes are available on the MXK, use the **show system** command.

```
zSH> show system
syscontact:-----> {260}
sysname:-----> {260}
syslocation:-----> {260}
enableauthtraps:-----> enabled disabled
setserialno:-----> {0 - 2147483647}
zmsexists:-----> true false
zmsconnectionstatus:--> active inactive
zmsipaddress:-----> {0 - 0}
configsyncexists:-----> true false
configsyncoverflow:---> true false
configsyncpriority:---> none low medium high
configsyncaction:-----> noaction createlist createfulllist
configsyncfilename:---> {68}
configsyncstatus:-----> synccomplete syncpending syncerror syncinitializing
configsyncuser:-----> {36}
configsyncpasswd:-----> {36}
numshelves:-----> {0 - 0}
shelvesarray:-----> {36}
numcards:-----> {0 - 0}
ipaddress:-----> {0 - 0}
alternateipaddress:---> {0 - 0}
countryregion:-----> argentina australia belgium china costarica finland france
germany hongkong italy japan korea mexico netherlands newzealand singapore spain
sweden switzerland uk us afghanistan albania algeria americansamoa andorra angola
anguilla antarctica antiguabarbuda armenia aruba austria azerbaijan bahamas
bahrain angladesh arbados elarus belize benin bermuda bhutan bolivia
bosniaherzegovina botswana bouvetisland brazil britishindianoceanterritory
bruneidarussalam bulgaria burkinafaso burundi cambodia cameroon canada capeverde
caymanislands centralafricanrepublic chad chile christmasisland cocosislands
colombia comoros congo cookislands cotedivoire croatia cuba cyprus czechrepublic
denmark djibouti dominica dominicanrepublic easttimor ecuador egypt elsalvador
equatorialguinea eritrea estonia ethiopia falklandislands faroeislands fiji
frenchguiana frenchpolynesia frenchsouthernterritories gabon gambia georgia ghana
gibraltar greece greenland grenada guadeloupe guam guatemala guinea guineabissau
guyana haiti heardislandmcdonaldislands holysee honduras hungary iceland india
indonesia iran iraq ireland israel jamaica jordan kazakstan kenya kiribati northkorea
kuwait kyrgyzstan lao latvia lebanon lesotho liberia libyanarabjamahiriya
liechtenstein lithuania luxembourg macau macedonia madagascar malawi malaysia
maldives mali malta marshallislands martinique mauritania mauritius mayotte
micronesia moldova monaco mongolia montserrat morocco mozambique myanmar namibia
nauru nepal netherlandsantilles newcaledonia nicaragua niger nigeria niue
norfolkisland northernmarianaislands norway oman pakistan palau palestinianterritory
panama papuanewguinea paraguay peru philippines pitcairn poland portugal puertorico
qatar reunion romania russia rwanda sainthelena saintkittsnevis saintlucia
saintpierreemiquelon saintvincentthegrenadines samoa sanmarino saotomeprincipe
```

```

saudiarabia senegal seychelles sierraleone slova kia slovenia solomonislands somalia
southafrica southgeorgia srilanka sudan suriname svalbardjanmayen swaziland syria
taiwan tajikistan tanzania thailand togo tokelau tonga trinidadtobago tunisia turkey
turkmenistan turkscaicosislands uganda ukraine unitedarabemirates uruguay uzbekistan
vanuatu venezuela vietnam virginislands uk virginislands us wallisfutuna
westernsahara yemen yugoslavia zambia zimbabwe
primaryclocksource:---> [Shelf {0-255}/Slot {0-31}/Port {0-500}/SubPort/Type] |
[Name/Type]
ringsource:-----> internalringsourcelabel externalringsourcelabel
revertiveclocksource:-> true false
voicebandwidthcheck:--> true false
alarm-levels-enabled:-> critical+major+minor+warning
userauthmode:-----> local radius radiusthenlocal
radiusauthindex:-----> {0 - 2147483647}
secure:-----> enabled disabled
webinterface:-----> enabled disabled

```

Use additional show commands such as **show bridge-interface-record** to view greater detail about bridges.

```

zSH> show bridge-interface-record
vpi:-----> {0 - 4095}
vci:-----> {0 - 65535}
vlanId:-----> {0 - 4090}
stripAndInsert:-----> false true
customARP:-----> false true
filterBroadcast:-----> false true
learnIp:-----> false true
learnUnicast:-----> false true
maxUnicast:-----> {0 - 2147483647}
learnMulticast:-----> false true
forwardToUnicast:-----> false true
forwardToMulticast:-----> false true
forwardToDefault:-----> false true
bridgeIfCustomDHCP:-----> false true
bridgeIfIngressPacketRuleGroupIndex:-> {0 - 2147483647}
vlanIdCOS:-----> {0 - 7}
outgoingCOSOption:-----> disable all
outgoingCOSValue:-----> {0 - 7}
s-tagTPID:-----> {33024 - 37376}
s-tagId:-----> {0 - 4090}
s-tagStripAndInsert:-----> false true
s-tagOutgoingCOSOption:-----> s-tagdisable s-tagall
s-tagIdCOS:-----> {0 - 7}
s-tagOutgoingCOSValue:-----> {0 - 7}
mcastControlList:-----> {264}
maxVideoStreams:-----> {0 - 210}
isPPPoA:-----> false true
floodUnknown:-----> false true
floodMulticast:-----> false true
bridgeIfEgressPacketRuleGroupIndex:-> {0 - 2147483647}
bridgeIfTableBasedFilter:-----> none+mac+ip
bridgeIfDhcpLearn:-----> none+mac+ip

```

get command

Use the **get** command to view the current configuration of profiles. The **get system 0** command displays information on the current MXK system configuration.

```

zSH> get system 0
system 0
Please provide the following: [q]uit.
syscontact: -----> {Zhone Global Services and Support 7001 Oakport
Street Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113
support@zhone.com}:
sysname: -----> {Zhone MxK}:
syslocation: -----> {Oakland}:
enableauthtraps: -----> {disabled}:
setserialno: -----> {0}:
zmsexists: -----> {false}:
zmsconnectionstatus: --> {inactive}:
zmsipaddress: -----> {0.0.0.0}:
configsyncexists: -----> {false}:
configsyncoverflow: ---> {false}:
configsyncpriority: ---> {high}:
configsyncaction: -----> {noaction}:
configsyncfilename: ---> {}:
configsyncstatus: -----> {syncinitializing}:
configsyncuser: -----> {}:
configsyncpasswd: -----> {** private **}: ** read-only **
numshelves: -----> {1}:
shelvesarray: -----> {}:
numcards: -----> {3}:
ipaddress: -----> {0.0.0.0}:
alternateipaddress: ---> {0.0.0.0}:
countryregion: -----> {us}:
primaryclocksource: ---> {0/0/0/0/0}:
ringsource: -----> {internalringsourcelabel}:
revertiveclocksource: -> {true}:
voicebandwidthcheck: --> {false}:
alarm-levels-enabled: -> {critical+major+minor+warning}:
userauthmode: -----> {local}:
radiusauthindex: -----> {0}:
secure: -----> {disabled}:
webinterface: -----> {enabled}:
options: -----> {NONE(0)}:

```

You can find the syscontact information, or whether the MXK is configured to communicate with the Zhone Management System (ZMS — zmsexists, zmsconnectionstatus, zmsipaddress).

update command

To update the **system 0** profile and all other profiles, use the **update** command. The **update system 0** command walks you through the profile to change specific fields.

Caution: You should be very careful when altering profiles. Where available you should use CLI macro commands.

For example:

```
zSH> update system 0
system 0
Please provide the following: [q]uit.
syscontact: -----> {Zhone Global Services and Support 7001 Oakport Street
Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113 support@zhone.com}:
sysname: -----> {Zhone MxK}:
syslocation: -----> {Oakland}:
enableauthtraps: -----> {disabled}:
setserialno: -----> {0}:
zmsexists: -----> {true>false
...
...
Save changes? [s]ave, [c]hange or [q]uit:
Record updated.
```

delete command

Use the **delete** command to delete profiles.

```
zSH> delete gpon-traffic-profile 1
gpon-traffic-profile 1
1 entry found.
Delete gpon-traffic-profile 1? [y]es, [n]o, [q]uit : y
gpon-traffic-profile 1 deleted.
```

Commands: interface show, host show, bridge show

This section describes these commands:

- [interface show command, page62](#)
- [host show command, page63](#)
- [bridge show command, page63](#)

interface show command

The **interface show** command displays the numbered or unnumbered (floating) IP interfaces currently available on the MXK.

```
zSH> interface show
1 interface
Interface      Status  Rd/Address          Media/Dest Address  IfName
-----
1/a/1/0/ip     UP      1 172.24.64.91/24    00:01:47:17:da:0e    ethernet1
-----
```

Table 5: Interface show column

Column	Description
Interface	Shows the interface, the card and the physical port of the IP interface.
Status	Shows whether the interface is up or down.
Rd/Address	The IP address assigned to this gateway.
Media/Dest Address	Media/Dest Address is either the MAC address of a device.
IfName	The interface name.

host show command

The **host show** command displays interfaces when the MXK is hosting a multi-point subnets.

```

zSH> host show
Rd/Address      Interface      Group T Host Address
-----
1 10.107.8.254  1-13-1-0-eth  1      D <unassigned>
               D <unassigned>
               D <unassigned>
               D <unassigned>
               D <unassigned>

```

bridge show command

The **bridge show** command displays the bridge interfaces on the MXK. Note that a bridge is a combination of bridge interfaces working together.

```

zSH> bridge show
Type VLAN      Bridge      St Table Data
-----
upl Tagged 2001 ethernet3-2001/bridge UP S VLAN 2001 default
upl Tagged 500 ethernet3-500/bridge UP S VLAN 500 default
dwn          500 1-4-1-0-eth/bridge UP
tls Tagged 101 ethernet2-101/bridge UP
dwn Tagged 500 1-7-7-501-gponport-500/bridge UP D 00:00:ff:00:00:01
               D 00:02:16:36:43:41
dwn Tagged 500 1-7-7-502-gponport-500/bridge UP D 00:00:ff:00:00:02
               D 00:02:16:36:43:42

```

Use the **bridge show** command with a VLAN ID to view all the bridges on a VLAN.

```

zSH> bridge show vlan500
Type VLAN      Bridge      St Table Data
-----
upl Tagged 500 ethernet3-500/bridge UP S VLAN 500 default
dwn          500 1-4-1-0-eth/bridge UP
dwn Tagged 500 1-7-7-501-gponport-500/bridge UP D 00:00:ff:00:00:01

```

```

dwn Tagged 500    1-7-7-502-gponport-500/bridge  UP  D 00:02:16:36:43:41
                                                           D 00:00:ff:00:00:02
                                                           D 00:02:16:36:43:42

```

Use the **bridge show** <bridge interface> command to view bridge interface information.

```
zSH> bridge show 1-4-1-501-gponport-160/bridge
```

```

Type VLAN          Bridge                               St  Table Data
-----

```

```

tls Tagged 160    1-4-1-501-gponport-160/bridge  UP

```

Commands: bridge stats

You can use the **bridge stats** command to view the packets being sent or received on bridge interfaces. If you add the name of a bridge you can see stats for that bridge.

```
zSH> bridge stats
```

Interface	Received Packets/Sec			Transmitted Packets/Sec			
Name	UCast	MCast	BCast	UCast	MCast	BCast	Error
ethernet3-2001	20101k	45635	94523	0	0	0	0
ethernet3-500	124M	0	239k	117M	0	317	0
1-4-1-0-eth	0	0	0	0	0	226k	0
ethernet2-101	0	0	0	0	0	0	0
1-7-7-501-gponport-500	--	--	--	3991k	0	224k	0
1-7-7-502-gponport-500	--	--	--	1783k	0	224k	0
1-7-7-503-gponport-500	--	--	--	1790k	0	224k	0
1-7-7-504-gponport-500	--	--	--	1787k	0	224k	0
1-7-7-505-gponport-500	--	--	--	3948k	0	224k	0

Commands: port show, port up, port down, port bounce

You can use the **port** command to view the administrative state of an interface, change the administrative state of an interface, or change configuration parameters for an interface.

Enter **port show** <interface> to view the administrative state of an interface:

```
zSH> port show 1-13-1-0/eth
```

```

Interface 1-13-1-0/eth
  Physical location:    1/13/1/0/eth
  Description:         510 505 5555
  Administrative status: up
  Port type specific information:
    Link state mirroring not configured.

```

Enter **port up** <interface> to change the administrative state of an interface from down to up:

```
zSH> port up 1-13-1-0/eth
```

```
1-13-1-0/eth set to admin state UP
```


Enter **port down** <interface> to change the administrative state of an interface from up to down:

```
zSH> port down 1-13-1-0/eth
1-13-1-0/eth set to admin state DOWN
```

Enter **port bounce** <interface> to change the interface from UP to DOWN, and back to UP.

```
zSH> port bounce 1-13-1-0/eth
1-13-1-0/eth set to admin state DOWN
1-13-1-0/eth set to admin state UP
```

Navigate the MXK file system

This section describes the MXK file system and includes:

- [Access the MXK file system, page65](#)
- [Download software files, page66](#)

Access the MXK file system

Use the following commands to access the MXK file system:

- **cd** Changes directory.
- **dir** Lists the contents of the directory.
- **pwd** Displays the current working directory.
- **image** Verifies software images and downloads software images on the flash to system memory.

The uplink card flash memory contains DOS file system that stores the system boot code, software images, and the configuration. During system startup, the software images on the flash are decompressed and loaded into memory.

Use the **cd**, **dir**, and **pwd** commands to list the contents of the file system, as in the following example:

```
zSH> cd /card1
```

```
zSH> pwd
/card1
```

```
zSH> dir
Listing Directory ..
-rwxrwxrwx  1 0      0      699784 Jan  7 10:50 mxup8graw.bin
-rwxrwxrwx  1 0      0     10089437 Jan  7 10:49 mxup8g.bin
-rwxrwxrwx  1 0      0     3803178 Jan  7 10:46 mxlc20ae.bin
drwxrwxrwx  1 0      0      4096 Jan 12 10:23 datastor/
drwxrwxrwx  1 0      0      4096 Nov 12  2008 onreboot/
drwxrwxrwx  1 0      0      4096 Jan  9 08:20 log/
drwxrwxrwx  1 0      0      4096 Oct 22  2008 bulkstats/
drwxrwxrwx  1 0      0      4096 Nov 24  2008 pub/
```

```

-rwxrwxrwx  1 0      0          5252736 Jan  7 10:47 mxlc4gp.bin
-rwxrwxrwx  1 0      0          694780 Jul 28 2008 mxup2tg8g.bin
-rwxrwxrwx  1 0      0          4977995 Dec  5 2008 mxlc8gp.bin
drwxrwxrwx  1 0      0           4096 Nov 25 2008 omci/
-rwxrwxrwx  1 0      0           748 Nov  3 2008 rsa.der
-rwxrwxrwx  1 0      0          1058 Nov  3 2008 rsakey.dat
-rwxrwxrwx  1 0      0         7742464 Jan  7 10:44 mxk819_http.tar
127381724 bytes available

```

Download software files

The MXK contains a TFTP client that enables you to download files from a network to the flash card file system using the **image** command. A software image for the uplink card and each type of line card must be downloaded.

The **image** command uses the following syntax:

image download tftphost imagefilename

The following example downloads the software image for the uplink card (*mxkup2tg8graw.bin*) from host 192.168.8.21 to the root directory of the first flash card:

image download 192.168.8.21 mxup2tg8graw.bin

Downloading software files

Download software files from the TFTP server to the MXK when you need to upgrade the system software:

- 1 Create the *onreboot* directory if one does not already exist and back up the current configuration file to the a file named *restore*, then **cd** back to the root directory.

```

zSH> mkdir onreboot
zSH> cd onreboot
zSH> dump file restore
zSH> cd ..

```

The *restore* file is used to restore the system configuration or revert to a previous release, if desired. See [Step 5](#).

- 2 Copy the new system boot image software to the flash memory using the **image download** command.

```
zSH> image download 192.168.8.21 mxup2tg8g.bin
```

where 192.168.8.21 is the TFTP server, and *mxup2tg8g.bin* is the name of the software image.

Caution: Be sure to download the correct software for the system.

- 3 Initialize the flash card's boot partition with the new image on both the primary and standby uplink card (if present).

For a single uplink card enter:

```
zSH> image flash mxup2tg8g.bin 1 1
```

For redundant uplink cards enter:

```
zSH> image flash mxup2tg8g.bin 1 all
```

- 4 The **image** command can also verify image files on the flash card. It reads the contents of the file, verifies the file header, and verifies the file checksum. For example:

```
zSH> image verify mxup2tg8g.bin
File: mxup2tg8gaw.bin
Size: 688320 bytes
Header Version: 1
Card Type: MX TWO TENGIGE EIGHT GIGE
Load Type: binary
Load Address: 0x00010000
Checksum: 0x2f66bb70
Image verify successful
```

The command reports any errors it finds in the file. Note that files are also verified as part of the download process.

- 5 Reset the system and restore the system configuration with the **systemreboot** command:

```
zSH> systemreboot
A restore file (/card1/onreboot/restore) is present.
A system reboot will result in a database restore.
Continue? (yes or no) [no]: yes
Do you want to reboot the system? (yes or no) [no] yes
Do you want to exit from this request? (yes or no) [yes] no
Are you sure? (yes or no) [no] yes
```

As shown above, when the *restore* file is present, the system displays

A restore file (/card1/onreboot/restore) is present.

and uses that file to restore the saved configuration to the MXK system.

After upgrading the software, the system automatically upgrades the software database to the new level.

Backup system configurations

The **dump** command saves the system configuration to the console, a local file, or the network.

The command uses the following syntax:

```
dump [file filename] [network host filename]
```

Passwords are encrypted when they are saved to the configuration file. The encrypted passwords are used to restore the correct password, but cannot be used to log in.



Note: The **dump** command uses TFTP to transfer files to the network. Set the TFTP server time-out value to at least 5 seconds, and 5 retries to help prevent TFTP timeout or retry errors.

Backing up the configuration to a local file

To dump the configuration to a local file:

Specify a file name for the configuration:

```
zSH> dump file filename
```

The file is saved on the MXK file system.

Backing up the configuration to the network

To back up the configuration to the network:

- 1 Create the file in the destination location of the TFTP server and make it writeable.
- 2 Save the configuration. The following example saves the configuration to a file named `device.cfg` on the host `192.168.8.21`:

```
zSH> dump network 192.168.8.21 device.cfg
```

Restoring the configuration

The configuration is restored to the system during **systemreboot**. See [Downloading software files on page 66](#).

SNTP

To set up the system to use SNTP update the **ntp-client-config** profile:

```
zSH> update ntp-client-config 0
ntp-client-config 0
Please provide the following: [q]uit.
primary-ntp-server-ip-address: ---> {0.0.0.0}192.168.8.100
secondary-ntp-server-ip-address: -> {0.0.0.0}:
local-timezone: -----> {gmt}pacific
daylight-savings-time: -----> {false}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

MXK Simple Network Management Protocol (SNMP)

This section describes the following:

- [Create SNMP community names and access profiles, page69](#)
- [Configure traps, page70](#)

Create SNMP community names and access profiles



Note: By default, the MXK has a single SNMP community defined with the name **ZhonePrivate**. This community has admin access to the system. Zhone recommends that you configure community names and access profiles to prevent unauthorized access to the system.

The **community-profile** specifies the community name and an access level for SNMP manager to access the system. It can also optionally specify a **community-access-profile** which is used to verify the source IP address of the SNMP manager. The system supports up to 50 different access lists.

The following community access levels are supported:

- **noaccess**—the community has no access.
- **read**—the community has read-only access to the system, with the exception of information in the **community-profile** and **community-access-profile**.
- **readandwrite**—the community has read/write access to the system, with the exception of information in the **community-profile** and **community-access-profile**.
- **admin**—the community has read and write access to the entire system, including information in the **community-profile** and **community-access-profile**. Note that the ZMS requires admin access to manage the system.

Create a community profile



Note: Configuring a community profile disables the **ZhonePrivate** default community name. If you do change the community name, you must change the name in ZMS or the device will become unmanageable.

The following example defines a community name **public** with read-only privileges:

```
zSH> new community-profile 1
Please provide the following: [q]uit.
community-name: -----> {}: public
```

```
permissions: -----> {read}:
access-table-index: -> {0}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Create community access profiles

The following example defines a community name **private** with read/write privileges and also creates an access list to verify that the SNMP client attempting to access the MXK is coming from known IP addresses 192.168.9.10 and 192.168.11.12:

First, create an access list for the first IP address:

```
zSH> new community-access-profile 2
Please provide the following: [q]uit.
access-table-index: -> {0}: 1
ip-address: -----> {0.0.0.0}: 192.168.9.10
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Then, create an access list for the second IP address with the same **access-table-index** (1):

```
zSH> new community-access-profile 3
Please provide the following: [q]uit.
access-table-index: -> {0}: 1
ip-address: -----> {0.0.0.0}: 192.168.11.12
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Finally, create a **community-profile** that specifies the community name, and uses the same **access-table-index** (1) as defined in the two **community-access-profiles** you just created:

```
zSH> new community-profile 4
Please provide the following: [q]uit.
community-name: -----> {}: private ZMS must include this name
permissions: -----> {read}: readandwrite
access-table-index: -> {0}: 1
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Configure traps

The **trap-destination** profile defines a trap recipient the MXK will send traps to. To configure a trap destination you need:

- the IP address of the SNMP trap server

- the community name the trap recipient expects

The other parameters in the **trap-destination** profile can be left at their default values. The following example configures a trap recipient with the IP address 192.168.3.21:

```
zSH> new trap-destination 32
Please provide the following: [q]uit.
trapdestination: -> {0.0.0.0}: 192.168.3.21
communityname: ---> {}: public
resendseqno: -----> {0}:
ackedseqno: -----> {0}:
traplevel: -----> {low}:
traptype: -----> {(null)}: 0
trapadminstatus: -> {enabled}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```



Note: When ZMS configures a device, a trap destination profile is automatically created.

Use log files to monitor the system

This section explains how to use logging on the MXK. It includes:

- [Overview, page71](#)
- [Enable/disable temporary logging sessions, page36](#)
- [Log message format, page72](#)
- [Modify logging levels, page74](#)
- [Non-persistent log messages, page75](#)
- [Persistent log messages, page77](#)
- [Example log messages, page77](#)
- [Log filter command, page78](#)
- [Send messages to a syslog server, page79](#)
- [Specify different log formats for system and syslog messages, page80](#)

Overview

Logging enables administrators to monitor system events by generating system messages. It sends these message to:

- A management session (either on the serial craft port or over a Telnet session)
- A log file on the device

- A syslog server (optional)

The type of information sent in these messages can be configured using the **log** command. By default, the system sends the same type of information to all log message destinations. If you want to send different types of messages to the syslog daemon, use the **syslog** command.

Enable/disable logging

By default, log messages are enabled on the serial craft port. Use the **log session** command and the **log serial** command to enable/disable logging:

The **log session** command enables/disables logging messages for that session only. If the user logs out, the logging setting returns to the default. To enable logging for the current session only:

```
zSH> log session on
```

To disable logging for the session:

```
zSH> log session off
```

The **log serial** command enables/disables logging messages for all sessions on the serial craft port. This setting persists across system reboots. To enable/disable logging for the serial craft port:

```
zSH> log serial on
```

To disable logging for the serial port:

```
zSH> log serial off
```

Log message format

Log messages contain the following information:

Table 6: Default log message fields

Option	Description
Date	Date stamp of log message. Enabled by default.
Time	Time stamp of log message. Enabled by default.
Ticks	Current tick count. When the tick option is used, the date and time fields are not displayed.
Level	Logging level of the message. Enabled by default.
Address	The shelf and slot and application identifier causing the alarm.
Logtest	Log handle.
Taskname	Name of task that generated the log message. This is generally useful only for Zhone development engineers. Enabled by default.
Function	Function that generated the log message.

Table 6: Default log message fields (Continued)

Option	Description
Line	Line in code that generated the log message. This is generally useful only for Zhone support staff.
Port	Port related to the log message.
Category	Category of the log message.
System	System related to the log message.
All	Controls all log message options.
Default	Controls the default log message options.
Message text	A description of the error that caused the alarm.

To change the information displayed in the log messages, use the **log option** command. First, display the available options:

zSH> **log option**

```
Usage: log option < time      | 1 > < on | off >
                  < date      | 2 > < on | off >
                  < level     | 3 > < on | off >
                  < taskname  | 4 > < on | off >
                  < taskid    | 5 > < on | off >
                  < file      | 6 > < on | off >
                  < function  | 7 > < on | off >
                  < line      | 8 > < on | off >
                  < port      | 9 > < on | off >
                  < category  | 10 > < on | off >
                  < system    | 11 > < on | off >
                  < ticks     | 12 > < on | off >
                  < stack     | 13 > < on | off >
                  < all       | 14 > < on | off >
                  < default   | 15 > < on | off >
```

option 'ticks' supercedes options 'time' & 'date'

time: date: level: address: log: taskname: function: line: port: category: system: (0x7cf)

Then, turn the option **on** or **off**. For example, the following command will turn the task ID off in log messages:

zSH> **log option taskid off**

time: date: level: address: log: taskname: (0xf)

The following commands will turn on/off the tick count display in log messages:

zSH> **log option ticks on**

time: date: level: address: log: port: category: system: ticks: (0xf07)

zSH> **log option ticks off**

time: date: level: address: log: port: category: system: (0x707)

The following command will turn all options on in log messages:

```
zSH> log option all on
time: date: level: address: log: taskname: taskid: file: function: line: port:
category: system: ticks: (0xffff)
```

Modify logging levels

To modify logging, use the **log** command. To modify syslog messages, use the **syslog** command.

Caution: Changing the log level may generate enough output to disrupt service.

To display the current levels for all logging modules, use the **log show** command:

```
zSH> log show
MODULE                                LEVEL      STATUS
adslhdlr                             error      enabled
adslprof                             error      enabled
alarm_mgr                            error      enabled
assert                               error      enabled
atm_cc_mib_hdlr                      error      enabled
atmmgragnt                           error      enabled
bds                                  error      enabled
bds_client                           error      enabled
bridge                               error      enabled
bridgemib                            error      enabled
bridgerp                             error      enabled
bulkstats                            error      enabled
bulkstatshdlr                       error      enabled
cam                                  error      enabled
card                                  error      enabled
card_resource                        error      enabled
carddeletehdlr                      info       enabled
cardred                              error      enabled
cardsvchdlr                         error      enabled
ccrp                                 error      enabled
ccrr                                 error      enabled
cesmibhdlr                          error      enabled
cli                                  error      enabled
clkmgr                              warning    enabled
....
```

Logging levels determine the number of messages that are displayed on the console. The higher the log level, the more messages are displayed. The MXK supports the following log levels:

- 1: emergency
- 2: alert

- 3: critical
- 4: error
- 5: warning
- 6: notice
- 7: information
- 8: debug

To change the log level, use the **log module level** command. For example, the following command changes the card module logging level to emergency:

Caution: Changing the log level may generate enough output to disrupt service.

```
zSH> log level card emergency
Module: card at level: emergency
```

To enable or disable log levels for a module, use the log enable or log disable commands. For example:

```
zSH> log disable card
Module: card is now disabled
```

Non-persistent log messages

The **log cache** command displays the non-persistent log cache messages:

```
zSH> log cache
[1]: MAY 19 14:28:31: alert   : 1/a/1025: alarm_mgr: 01: a:06 Critical ETHERNET Down -
Ethernet line down
[2]: MAY 19 14:30:19: alert   : 1/13/1025: alarm_mgr: 01:13:01 Major ETHERNET Up -
Ethernet line up
[3]: MAY 19 14:32:12: alert   : 1/13/1025: alarm_mgr: 01:13:01 Major ETHERNET Down -
Ethernet line down
[4]: MAY 19 14:32:26: alert   : 1/13/1025: alarm_mgr: 01:13:02 Major ETHERNET Up -
Ethernet line up
[5]: MAY 19 14:33:27: alert   : 1/13/1025: alarm_mgr: 01:13:02 Major ETHERNET Down -
Ethernet line down
[6]: MAY 19 14:36:23: alert   : 1/4/1025: alarm_mgr: 01: 4:01:01 Minor ONU Down
Line 1/4/1/1/gpononu CAUSE: inactive
[7]: MAY 19 14:36:32: alert   : 1/4/1025: alarm_mgr: 01: 4:01:01 Minor ONU Up
Line 1/4/1/1/gpononu CAUSE: active
[8]: MAY 19 14:36:53: critical: 1/a/1035: rebootserver:
* * * * Slot Reboot : type = 2, shelf = 1, slot = 4
[9]: JAN 01 00:00:11: error   : 1/4/9    : tnettask: Unable to find ifnet pointer for
ifindex 0x2c0
[10]: JAN 01 00:00:11: error   : 1/4/9    : tnettask: Unable to find ifnet pointer for
ifindex 0x2c1
[11]: JAN 01 00:00:12: error   : 1/4/9    : tnettask: Unable to find ifnet pointer for
ifindex 0x2c2
```

```
[12]: MAY 19 14:40:32: notice : 1/a/12 : shelfctrl: Card in slot 4 changed state to RUNNING.
```

```
[14]: MAY 19 14:40:32: alert : 1/4/1025: alarm_mgr: 01: 4:02 Critical OLT Up
```

```
Line 1/4/2/0/gponolt CAUSE: active
```

The **log cache max length** command sets the maximum number of log messages to store. The maximum log cache size is 2147483647, depending in the amount of memory available.

log cache max length

To change the current configured log cache size:

```
zSH> log cache max 200
```

```
Maximum number of log messages that can be saved: 200
```

The **log cache grep pattern** command searches through the log cache for the specified regular expression.

log cache grep pattern

The following example searches through the log cache for the string "Critical":

```
zSH> log cache grep Critical
```

```
Searching for: "Critical"
```

```
[1]: AUG 02 22:37:19: alert : 1/a/1025: alarm_mgr: 01: a:01 Critical ETHERNET Up - Ethernet line up
```

```
[2]: AUG 02 22:37:34: alert : 1/a/1025: alarm_mgr: 01: a:02 Critical ETHERNET Down - Ethernet uplink down
```

```
[3]: AUG 02 22:37:34: alert : 1/a/1025: alarm_mgr: 01: a:03 Critical ETHERNET Down - Ethernet line down
```

The **log cache clear** command clears the log cache.

log cache clear

The **log cache size** command sets the maximum amount of memory for the log cache. Without options, displays the current log size.

```
zSH> log cache size
```

```
Number of log messages in the cache: 20
```

```
Total bytes used by the cache: 2052
```

The **log cache help** command displays the help information for the **log cache** command:

```
zSH> log cache help
```

```
Usage: log cache < max > < length >
```

```
< grep > < pattern >
```

```
< clear >
```

```
< size >
```

```
< help >
```

With no arguments the 'log cache' command prints out all the

log messages currently in the cache.

The 'max' command is used to view/set the maximum number of log messages that can be cached at one time. If the cache is full then the oldest log is discarded and the new log is inserted. If no value is given then the current setting is displayed

The 'size' command is used to display the amount of memory currently being used by the log cache.

The 'clear' command is used to erase the log cache.

The 'grep' command is used for searching the log cache for a specific pattern. Extended regular expressions are supported.

Persistent log messages

Use the **log cache** command to view the persistent logs which only stores emergency level logs. For example:

```
zSH> log display
```

```
JAN 08 18:11:41: emergency: 1/a/12 : shelfctrl: Critical alarm set!
JAN 09 08:16:04: emergency: 1/a/12 : shelfctrl: Critical alarm set!
JAN 12 10:25:26: emergency: 1/b/12 : shelfctrl: Critical alarm set!
```

Example log messages

This section provides examples of how to interpret log messages.

Card line down message

The following message appears when a card in the MXK chassis comes up or goes down.

The most important parts of the message are the date and time the event occurred, the shelf/slot of the event, and the message text. The remainder of the information is only useful for Zhone development engineers.

Date and time	Log level	Physical address	Task name
[1]: MAY 19 14:28:31:	alert:	1/a/1025:	alarm_mgr: 01: a:06
Critical ETHERNET Down - Ethernet line down			

Message text

Line card up message

The next message appears after a line card has finished loading its software and is ready to be provisioned.

The most important parts of the message are the date and time the event occurred, the physical address (shelf/slot) of the event, and the message text.

Date and time	Log level	Physical address	Task name
[4]: APR 07 11:52:01: notice: 1/a/12: shelfctrl: _CardUpdateMsgProcess(): l=491: tShelfCtrl: Card in slot 13 changed state to RUNNING			
Message text			

Log filter command

The **log filter** command is available as part of the log command functionality. This command enables users to show, set and delete log filters. Log filters limit the scope of log messages to a specific entity for troubleshooting and diagnostics. When a log filter is set, the filter is assigned an index number and only messages relate the specified entity are displayed. Filters can be set for an specific ifindex, slot/port, VCL, or subscriber.

log filter

Restrict the display of log messages to only the log messages for a specified entity.

Syntax `log filter show | set (ifindex|port slotport|vcl ifindex vpi vci|subscriber endpoint)| delete`

zSH> **log filter set ifindex 12**
New filter saved.

zSH> **log filter set port 5 24**
New filter saved.

zSH> **log filter set subscriber 22**
New filter saved.

zSH> log filter show		
Index	Type	Filter Parameters

1	Port	slot=1, port=1
2	Port	slot=1, port=4
3	IfIndex	IfIndex=12
4	Port	slot=5, port=24
6	IfIndex	IfIndex=100
7	IfIndex	IfIndex=104
8	IfIndex	IfIndex=109
9	IfIndex	IfIndex=103
10	IfIndex	IfIndex=107

zSH> **log filter delete 10**

Log filter 10 deleted

Send messages to a syslog server

[Table 7](#) describes the parameters in the **syslog-destination** profile you can modify to send messages to a syslog server.

Table 7: syslog-destination profile parameters

Parameter	Description
address	The IP address of the machine hosting the syslog server. Default: 0.0.0.0
port	The UDP port to which the syslog messages will be sent. Default: 514
facility	The syslog facility to which the syslog messages will be sent. Values: local0 local1 local2 local3 local4 local5 local6 local7 no-map Default: local0
severity	The severity level used to filter messages being sent to the syslog server. Values: emergency alert critical error warning notice info debug Default: debug

```

zSH> new syslog-destination 1
Please provide the following: [q]uit.
address: --> {0.0.0.0}: 192.200.42.5 IP address of the syslog server
port: -----> {514}: leave at default
facility: -> {local0}:
severity: -> {debug}:

```

.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

Specify different log formats for system and syslog messages

Table8 describes the **log-module** profile that supports the configuration of persistent log messages, syslog messages, and persistent storage levels by module. Modify this profile when you need to send different messages to admin sessions, the persistent logs, and the syslog server.

Table 8: log-module profile parameters

Parameter	Description
name	The name of the module whose logging is controlled by this profile. Default: logtest
display	Controls the display of messages on the system. Messages logged at this level and above will be displayed. Values: emergency alert critical error warning notice info debug Default: error

Table 8: log module profile parameters (Continued)

Parameter	Description
syslog	Controls the format of messages sent to the syslog server described in the syslog-destination profile. This field is similar to the display field, except for the trackdisplay value. Values: emergency alert critical error warning notice info debug trackdisplay Messages logged at, and above, the level set in the display parameter will also be recorded in the syslog server. Default: trackdisplay
store	Controls the persistent storage of messages. This field is similar to the display field, except for the trackdisplay value. Values: emergency alert critical error warning notice info debug trackdisplay Messages logged at, and above, the level set in the display parameter will also be recorded in the syslog server. Default: trackdisplay

```

zSH> new log-module 1
Please provide the following: [q]uit.
name: ----> {logtest}: test1
display: -> {error}: warning
syslog: --> {trackdisplay}:
store: ---> {trackdisplay}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

```

In this case, the **log-module 1** will display to the screen, all messages at and above **warning**. The variable **trackdisplay** means that the same messages as defined in **display** are also sent to the syslog and storage. If different level of

messages are needed for the different destinations, the variables for **display**, **syslog**, and **store** can be set at different levels.

Map subscriber information to a port description field on the MXK

This section describes port descriptions:

- [Port description rules, page82](#)
- [Add, modify, list, and delete a port description, page82](#)
- [Search port descriptions, page86](#)

Port description rules

The MXK has a port description field, which provides a mapping between the physical port, or bonded interface, or bridge and a subscriber. This mapping improves MXK management without requiring extra documents to provide the mapping. Port description information can be entered for ports, bridges, or bond groups. Port description information is also searchable.

The rules for entering a port description are:

- Port descriptions do not have to be unique.
- The port description field is a text field 64 characters long.
- Any characters can be used including spaces, \$, @, -, ., etc. The only characters not supported are the double quote, ‘ “ ‘which is a delimiter to identify the beginning and end of the text string, the carat ‘^’, and the question mark ‘?’.
- Port descriptions are associated with physical ports and not logical interfaces. For bonding technologies port descriptions are supported both on the physical port and the bond group, so if you want to use a keyword such as a company name to group interfaces.
- Even though port descriptions are searchable, you cannot perform commands using port description. For example, you can not use a command like “bridge modify circuitName ...”

Add, modify, list, and delete a port description

The **port description add** command associates a text string with a physical interface (which includes bond groups):

```
port description add <physical interface> <text string>
```



Note: Port descriptions do not need to be unique. If one customer has many lines, they may all have the same port description. You may also use the port description field as a means to group interfaces. See [Search port descriptions, page86](#).

Add a port description to a port

To add a port description with spaces to a port, enter:

```
zSH> port description add 1-13-1-0/eth "510 555 5555"
```

In this case, the port description has spaces so quotes are needed.

To verify the port description, enter:

```
zSH> port show 1-13-1-0/eth
Interface 1-13-1-0/eth
  Physical location:    1/13/1/0/eth
  Description:         510 555 5555
  Administrative status: up
  Port type specific information:
    Link state mirroring not configured.
```

To add a port description without spaces to a port, enter:

```
zSH> port description add 1-13-4-0/eth Carringtons
```

To verify the port description enter:

```
zSH> port show 1-13-4-0/eth
Interface 1-13-4-0/eth
  Physical location:    1/13/4/0/eth
  Description:         Carringtons
  Administrative status: up
  Port type specific information:
    Link state mirroring not configured.
```

Add a port description to a GPON port

GPON ports have one ONT and up to 64 ONUs. Both the ONT and the ONUs can have port descriptions.

To add a port description on a GPON ONT, enter:

```
zSH> port description add 1-4-1-0/gponolt SFO
```

To verify the port description, enter:

```
zSH> port show 1-4-1-0/gponolt
Interface 1-4-1-0/gponolt
  Physical location:    1/4/1/0/gponolt
  Description:         SFO
  Administrative status: up
```

To add a port description to a GPON ONU, enter:

```
zSH> port description add 1-4-1-1/gpononu "business 1 555-555-5555"
```

In this case, a port description is added to ONU 1 on OLT 1.

To verify the port description, enter:

```
zSH> port show 1-4-1-1/gpononu
Interface 1-4-1-1/gpononu
      Description:      business 1 555-555-5555
      Administrative status: up
```

Add a port description to a bridge

The port description must be added to the physical port of a bridge configuration. A port description can be added to the physical port of an existing bridge configuration or the port description can be added to the physical port that is then configured as a bridge.

View existing bridges:

```
zSH> bridge show
Type VLAN      Bridge      St  Table Data
-----
upl Tagged 200  ethernet8-200/bridge  DWN S VLAN 200 default
dwn Untagged 1-13-2-0-eth/bridge  DWN
```

Add the port description to the physical port of an existing bridge configuration, in this case the uplink bridge on Ethernet port 2:

```
zSH> port description add 1-13-2-0/eth "US Insurance Consortium, Inc."
```

Verify the port description on the uplink bridge:

```
zSH> bridge showdetail 1-13-2-0-eth/bridge
Bridge interface: 1-13-2-0-eth
      Administrative status: up      Operational status: down
      Type:dwn      Untagged      Data:
Physical interface: 1-13-2-0/eth
      Administrative status: up      Operational status: Line Down
Description: US Insurance Consortium, Inc.
Interface On Demand Stats State: disabled
Total Packet Statistics
      Received
            Unicast: 0
            Multicast: 0
            Broadcast: 0
      Sent
            Unicast: 0
            Multicast: 0
            Broadcast: 0
            Errors: 0
Delta Packet Statistics - Collecting a 1 second data interval
      Received      Sent
```

	Unicast	Multicast	Broadcast	Unicast	Multicast	Broadcast	Error
Delta	0	0	0	0	0	0	0
Rate	0	0	0	0	0	0	0

Add a port description to a bond group

View the existing bond groups:

```
zSH> bond show all
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
5	124	efmbond	00S	1-5-124-0	-
5	25	efmbond	00S	bond-0025	-

To add a port description to an existing bond group enter:

```
zSH> port description add bond-0025/efmbond "Mary's Nail Shop"
```

To verify the port description on the bond group enter:

```
zSH> bond show group bond-0025/efmbond
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
5	25	efmbond	00S	bond-0025	Mary's Nail Shop
Group Members					
Slot	Port	Type	State	Name	Desc
5	2	shdsl	00S	1-5-2-0	-
5	4	shdsl	00S	1-5-4-0	-
5	3	shdsl	00S	1-5-3-0	-

Or enter:

```
zSH> bond show all
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
5	124	efmbond	00S	1-5-124-0	-
5	25	efmbond	00S	bond-0025	Mary's Nail Shop

Modify a port description

The **port description modify** command allows you to edit an existing port description.

```
port description modify <physical interface> <text string>
```

Enter a port description:

```
zSH> port description add 1-4-1-2/gpononu "Cafe Roma"
```

Verify the description:

```
zSH> port show 1-4-1-2/gpononu
```

```
Interface 1-4-1-2/gpononu
  Description:      Cafe Roma
  Administrative status: up
```

Modify the description on the same port:

```
zSH> port description modify 1-4-1-2/gpononu "Cafe Barrone"
```

Verify the change:

```
zSH> port show 1-4-1-2/gpononu
Interface 1-4-1-2/gpononu
  Description:      Cafe Barrone
  Administrative status: up
```

Port description list

The **port description list** command will list the descriptions on a particular port.

```
zSH> port description list 1/4/1
```

Interface	Description
1-4-1-0/gponolt	SF0
1-4-1-1/gpononu	business 1 555-555-5555
1-4-1-2/gpononu	Cafe Barrone

Port description delete

The **port description delete** command removes the port description from the physical interface.

```
port description delete <physical interface>
```

To view the port description on a physical port enter:

```
zSH> port show 1-9-22-0/adsl
Interface 1-9-22-0/adsl
  Physical location: 1/9/22/0/adsl
  Description:      Cafe Barrone
  Administrative status: up
```

To delete the port description enter:

```
zSH> port description delete 1-9-22-0/adsl
```

To verify the deletion enter:

```
zSH> port show 1-9-22-0/adsl
Interface 1-9-22-0/adsl
  Physical location: 1/9/22/0/adsl
  Administrative status: up
```

Search port descriptions

The **port description find** command provides a textual search which allows you search for a text string within the port description fields. The display

show the description and the physical location. If multiple port descriptions have the same text string they will all be displayed

port description find <text string>

```
zSH> port description find 510
```

Results for 510

Description: 510 555 5555

Interface: 1-13-1-0/eth

```
zSH> port description find "business 1 555-555-5555"
```

Results for business 1 555-555-5555

Description: business 1 555-555-5555

Interface: 1-4-1-1/gpononu



Note: Notice that for search items which do not have spaces the quotation marks are unnecessary.

MXK security

This section describes the MXK's security features including Radius support, Secure Shell (SSH), Secure File Transfer Protocol (SFTP), HTTPS and port access security.

- [MXK security \(SSH, SFTP, and HTTPS\), page87](#)
- [Port access security, page91](#)
- [Radius support, page94](#)



Note: For security reasons, host keys are not accessible via SNMP and cannot be saved/restored with the dump command.

MXK security (SSH, SFTP, and HTTPS)

This section covers the security on the MXK:

- [Enable security on the MXK, page87](#)
- [DSA and RSA keys, page89](#)
- [Tested MXK SSH clients, page90](#)
- [Encryption-key commands, page90](#)

Enable security on the MXK

The **system 0** profile provides a **secure** parameter which allows only secure communication for management activities. When security is enabled, the MXK uses the following protocols:

- Secure File Transfer Protocol (SFTP)
- Secure shell (SSH)
- HTTPS (HTTP secure)

[Table 9](#) describes which protocols are allowed when the **secure** parameter is *enabled* and which protocols are allowed when the **secure** parameter is *disabled*.

Table 9: Protocols for the secure parameter

Disabled	Enabled
TFTP, FTP	SFTP
Telnet, SSH	SSH
HTTP	HTTPS

Enabling security on the MXK

- 1 To enable the security parameter enter **update system 0** on the MXK, change the secure parameter from *disabled* to *enabled*, then save the file:



Note: After enabling the **secure** parameter, HTTPS and changes to the Web UI take affect after the next reboot. SSH and SFTP do not require a reboot.

```

zSH> update system 0
system 0
Please provide the following: [q]uit.
syscontact: -----> {Zhone Global Services and Support 7001 Oakport
Street Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113
support@zhone.com}:
sysname: -----> {Zhone MxK}:
syslocation: -----> {Oakland}:
enableauthtraps: -----> {disabled}:
setserialno: -----> {0}:
zmsexists: -----> {false}:
zmsconnectionstatus: --> {inactive}:
zmsipaddress: -----> {0.0.0.0}:
configsyncexists: -----> {false}:
configsyncoverflow: ---> {false}:
configsyncpriority: ---> {high}:
configsyncaction: -----> {noaction}:
configsyncfilename: ---> {}:
configsyncstatus: -----> {syncinitializing}:
configsyncuser: -----> {}:
configsyncpasswd: -----> {** private **}: ** read-only **
numshelves: -----> {1}:
shelvesarray: -----> {}:

```



```

numcards: -----> {3}:
ipaddress: -----> {0.0.0.0}:
alternateipaddress: ---> {0.0.0.0}:
countryregion: -----> {us}:
primaryclocksource: ---> {0/0/0/0/0}:
ringsource: -----> {internalringsource}:
revertiveclocksource: -> {true}:
voicebandwidthcheck: --> {false}:
alarm-levels-enabled: -> {critical+major+minor+warning}:
userauthmode: -----> {local}:
radiusauthindex: -----> {0}:
secure: -----> {disabled}: enabled
webinterface: -----> {enabled}:
options: -----> {NONE(0)}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

DSA and RSA keys

The MXK automatically creates a Digital Signature Algorithm (DSA), a standard for digital signatures, and supports RSA, an algorithm for public-key cryptography. The DSA and RSA host keys for the server are persistently stored in the encryption-key profile. In order to manage the host keys, use the CLI command **encryption-key**.

RSA involves a public key and a private key. The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted using the private key.

When the system first boots, it will try to load the existing DSA and RSA keys. If they do not exist, the system creates a 512 bit DSA key.

The CLI **encryption-key** command can be used to view current keys, create a new key, regenerate keys that may have been compromised, and delete keys.

To create a new key enter:

```

zSH> encryption-key add rsa 1024
Generating key, please wait ... done.

```



Note: Generating keys is computationally intensive. The longer the key, the longer it takes to generate. Wait until the system shows that key generation is completed before you continue.

To view the new key just created enter:



Note: The **encryption-key show** command displays the keys that were generated and are available for use. The command does not show the actual keys.

```

zSH> encryption-key show
Index Type          Length

```

```
-----
1   dsa           512
2   rsa          1024
```

To regenerate a key that might have been compromised enter:

```
zSH> encryption-key renew dsa
Generating key, please wait ... done.
```

To delete an encryption key enter:

```
zSH> encryption-key delete dsa
```

Tested MXK SSH clients

Secure Shell (SSH) is a command interface and protocol for securely getting access to a remote computer. SSH commands are encrypted and secure in two ways. Both ends of the client/server connection are authenticated using a digital certificate, and passwords are protected by being encrypted. You can now connect to a MXK using the SSH client of your choice to encrypt the session. The MXK SSH2 only with the following SSH clients:

- OpenSSH
 - cygwin
 - Linux
 - Solaris
- Putty
- Teraterm
- SecureCRT
- Absolute Telnet

Encryption-key commands

encryption-key add

Adds an encryption key to the encryption-key profile.

Syntax `encryption-key add [rsa|dsa] [512|768|1024|2048]`

Options `rsa|dsa`

Name and type of the encryption key.

512|768|1024|2048

The number of bytes the key is set to.

encryption-key delete

Deletes an encryption key from the encryption-key profile.

Syntax `encryption-key delete [rsa|dsa]`

Options `rsa|dsa`
Name and type of the encryption key.

encryption-key renew

Regenerates a compromised encryption key.

Syntax `encryption-key renew [rsa|dsa]`

Options `rsa|dsa`
Name and type of the encryption key.

encryption-key show

Displays the current encryption keys.

Syntax `encryption-key show`

Port access security

The MXK provides security capabilities on the UDP/TCP ports which the MXK uses for management. Use the **port-access** profile to define the UDP/TCP port and the IP address or IP address subnet that allows access to that port.

The port access security feature is a white list mechanism. If a host's IP address is not specified in a **port-access** profile, users from that host cannot access on that port.

The management ports are:

- Telnet, port 23
- SSH, port 22
- HTTP, port 80
- HTTPS, port 443
- SNMP, port 161

In order to restrict access to the SNMP port, there must be a rule to allow the MXK its own SNMP access. See [Creating a port-access entry for the MXK to maintain SNMP access on page 93](#).

By default, **port-access** profiles do not exist and all ports are open. After a **port-access** profile is configured for a port all other IP addresses or subnets are blocked. This restriction only takes effect after the first **port-access** profile is created.

 **Note:** Port access security is not independent from enabling secure mode for SFTP and SSH in **system 0**. If secure is enabled to provide SSH and SFTP while limiting Telnet access, and you have provided access with the **port-access** profile for Telnet to a device (or range of devices), the device(s) will not have access.

Up to 100 **port-access** profile entries can be created on a SLMS device.

Creating port-access profile entries

Create a **port-access** profile entry.

- 1 Create a new port-access entry by entering **new port-access n**, where n is an available entry ID number.
- 2 In the **portNumber** parameter enter the port number.
- 3 In the **srcAddr** parameter enter the IP address or first IP address of the subnet.
- 4 In the **netMask** parameter enter 255.255.255.255 for a single IP address mask, or a subnet mask for a subnet.

Creating a port-access entry for a specific IP address

Create a new **port-access** profile and specify the port number, host/network IP address to be granted access, and the one address netmask (255.255.255.255, which really means an exact mask of the IP address given) applied to the IP address to allow access to a single IP address.

 **Note:** To create port access protection for both HTTP and HTTPS, create port access entries for port 80 and port 443.

```
zSH> new port-access 1
Please provide the following: [q]uit.
portNumber: -> {0}: 80
srcAddr: ---> {0.0.0.0}: 172.16.42.1
netMask: ---> {0.0.0.0}: 255.255.255.255
.....S=
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

This example creates **port-access** entry 1 on HTTP port 80 and allows hosts on the 172.16.42.1 network to have HTTP access to the MXK.

Creating a port-access entry for a subnet

Create a new **port-access** profile and specify the Telnet port number, initial host/network IP address to be granted access, and the netmask applied to the IP address to allow access to a range of IP addresses.



Note: Typically, only port 23 is used for Telnet access.

```
zSH> new port-access 2
Please provide the following: [q]uit.
portNumber: -> {0}: 23
srcAddr: ---> {0.0.0.0}: 172.16.41.0
netMask: ---> {0.0.0.0}: 255.255.255.0
.....S=
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

This example creates **port-access** entry 2 on Telnet port 23 and allows hosts on the 172.16.41.xx network to Telnet to the MXK.

Displaying port-access profile entries

Display configured **port-access** profile entries with the **list** command:

```
zSH> list port-access
port-access 1
1 entry found.
```

Modifying port-access profile entries

Modify a configured **port-access** profile entry with the **update** command.

This example changes the entry's source IP address to 172.16.40.0:

```
zSH> update port-access 2
portNumber: -> {23}
srcAddr: ---> {172.16.41.0} 172.16.40.0
netMask: ---> {255.255.255.0}
1 entry found.
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Updated record saved.
```

Displaying port-access profile entries

To display configured **port-access** profile entries use the **list** command:

```
zSH> list port-access
port-access 1
1 entry found.
```

Creating a port-access entry for the MXK to maintain SNMP access

Create a new **port-access** profile and specify the SNMP port number (161) then 127.0.0.0 as the IP address for the subnet and a subnet mask of 255.0.0.0.

```
zSH> new port-access 10
Please provide the following: [q]uit.
portNumber: -> {0}: 161
srcAddr: ---> {0.0.0.0}: 127.0.0.0
netMask: ---> {0.0.0.0}: 255.0.0.0
.....S=
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Radius support

The MXK supports local and RADIUS (Remote Authentication Dial In User Service) access authentication. The MXK can be configured for local authentication, RADIUS authentication, or RADIUS then local authentication. RADIUS users are configured with the Service-Type attribute as Administrative-User or NAS-Prompt-User. RADIUS is used for only login authentication, not severity levels.

Table10 shows the mapping of service-type to MXK permissions.

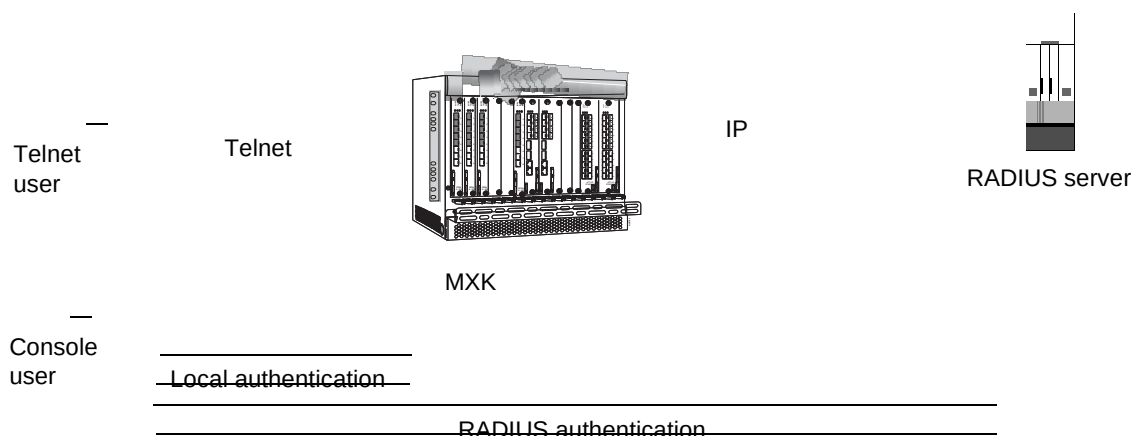
Table 10: Service type mapping to MXK permissions

Service-Type Attribute	MXK permissions
Administrative-User	admin, zhonedebug, voice, data, manuf, database, systems, tools, useradmin
NAS-Prompt-User	admin, voice, data, manuf, database, systems, tools, useradmin

When establishing a connection to the MXK with RADIUS authentication, the MXK passes RADIUS information securely to the RADIUS server. The RADIUS server then authenticates the user and either allows or denies access to the MXK. If access is denied and the local authentication option is also configured, the MXK then authenticates access based on the locally configured users and passwords. For logins and failed logins, a console message is generated with user ID and IP address of the device from which the login originated. Failed logins also are logged as alert level messages in the MXK system log file.

By default, RADIUS access uses the UDP port 1812 for authentication. This parameter can be changed in the radius-client profile.

Figure 5: MXK RADIUS authentication



Note: Follow the RADIUS server guidelines for RADIUS configuration instructions. For example, when using the MXK with the FreeRadius server:

- Create only one entry in the *clients.conf* file for each subnet or individual MXK. For individual MXKs, the IP in this file must match the IP address of the outbound interface used by the MXK to connect to the RADIUS server.
- The MXK uses the value stored in the RADIUS *system.sysname* file for the NAS-Identifier attribute.
- The shared-secret in the MXK **radius-client** profile, must exactly match the shared-secret in the RADIUS client entry.

Configuring RADIUS support

The MXK can be configured for local authentication, RADIUS authentication, or RADIUS then local authentication. Multiple **radius-client** profiles can be defined using the index and subindex numbers. This index scheme can be used to create index numbers for groups of RADIUS servers. When an index number is specified in the system profile, the MXK attempts authentication from each RADIUS server in that group in sequential order of the subindex numbers.

To configure RADIUS support:



Note: Before beginning this procedure, ensure that the MXK has IP connectivity to the RADIUS server.

- 1 Update the RADIUS server with settings for the Zhone prompts.
- 2 Create a **radius-client** profile on the MXK with the desired index number and RADIUS settings for server name, shared secret, number of retries, and other parameters. The first number in the index is used to group **radius-client** profiles so multiple profiles can be assigned to a MXK. The

second number in the index specifies the order in which **radius-client** profiles are referenced. This example specifies the **radius-client 1/1** with server name *radius1* and a **shared-secret** of *secret*. A DNS resolver must be configured in the system to resolve the server name and IP address. If a DNS resolver is not available, specify the IP address of the server. The index *1/1* specifies that this profile is the first profile in group *1*.

```
zSH> new radius-client 1/1
Please provide the following: [q]uit.
server-name: ----> {}: radius1.test.com [DNS resolver must be configured in the system.]
udp-port: -----> {1812}:
shared-secret: --> {** password **}: secret
retry-count: ----> {5}:
retry-interval: -> {1}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.
```

Another method to reference the RADIUS server is by specifying the IP address. This example specifies the **radius-client 1/1** with server IP address 172.24.36.148 and a **shared-secret** of *secret*. The index *1/1* specifies that this profile is the first profile in group *1*.

```
zSH> new radius-client 1/1
Please provide the following: [q]uit.
server-name: ----> {}: 172.24.36.248
udp-port: -----> {1812}:
shared-secret: --> {** password **}: secret
retry-count: ----> {5}:
retry-interval: -> {1}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.
```

- 3 Create another **radius-client** profile on the MXK with the desired RADIUS settings for server name, shared secret, number of retries, and other parameters. This example specifies the **radius-client 1/2** with server IP address 172.24.36.148 and a **shared-secret** of *secret*. The index *1/2* specifies that this profile is the second profile in group *1*.

```
zSH> new radius-client 1/2
Please provide the following: [q]uit.
server-name: ----> {}: 172.24.36.249
udp-port: -----> {1812}:
shared-secret: --> {** password **}: secret
retry-count: ----> {5}:
retry-interval: -> {1}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.
```

Create additional **radius-client** profiles for each additional RADIUS server to be assigned to this MXK.

- 4 In the system profile on the MXK, set the desired user authentication method and specify the index of the radius profile to use. This example specifies the **radiusauthindex** of 1. This index is configured with two **radius-client** profiles (1/1, 1/2). The MXK first attempts authentication using the server specified in **radius-client 1/1**. If this authentication fails, the MXK attempts authentication using **radius-client 1/2** server. If this authentication also fails, the MXK then attempts authentication based on the authentication mode setting in the system profile. This example uses **radiusthenlocal**.

Caution: If the *radius* authentication mode is used, local authentication is disabled so the MXK may become inaccessible if IP connectivity to the RADIUS server is lost or other changes prevent the MXK from receiving RADIUS authentication.

```

zSH> update system 0
system 0
Please provide the following: [q]uit.
syscontact: -----> {Zhone Global Services and Support 7001 Oakport
Street Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113
support@zhone.com}:
sysname: -----> {Zhone MxK}:
syslocation: -----> {Oakland}:
enableauthtraps: -----> {disabled}:
setserialno: -----> {0}:
zmsexists: -----> {false}:
zmsconnectionstatus: --> {inactive}:
zmsipaddress: -----> {0.0.0.0}:
configsyncexists: -----> {false}:
configsyncoverflow: ---> {false}:
configsyncpriority: ---> {high}:
configsyncaction: -----> {noaction}:
configsyncfilename: ---> {}:
configsyncstatus: -----> {syncinitializing}:
configsyncuser: -----> {}:
configsyncpasswd: -----> {** private **}: ** read-only **
numshelves: -----> {1}:
shelvesarray: -----> {}:
numcards: -----> {3}:
ipaddress: -----> {0.0.0.0}:
alternateipaddress: ---> {0.0.0.0}:
countryregion: -----> {us}:
primaryclocksource: ---> {0/0/0/0/0}:
ringsource: -----> {internalringsourcelabel}:
revertiveclocksource: -> {true}:
voicebandwidthcheck: --> {false}:
alarm-levels-enabled: -> {critical+major+minor+warning}:
userauthmode: -----> {local}: radiusthenlocal
radiusauthindex: -----> {0}: 1
secure: -----> {disabled}:
webinterface: -----> {enabled}:
options: -----> {NONE(0)}:

```

```
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

After completing the RADIUS configuration, the MXK displays console messages for RADIUS login and logout activity.

For users logging in through RADIUS, the system prompt appears as the *username@systemname*. For example, the system prompt for a basic user on a MXK using the default Zhone MXK system name will appear as *basicuser@Zhone mxk*. The system name is configured using the **sysname** parameter in the **system 0** profile.

MXK clocking

This section discusses:

- [View clocking source on the MXK, page98](#)
- [Set the TAC card as clocking source, page99](#)

View clocking source on the MXK

Default clocking for the MXK system, is located on the active uplink card. In cases where a synchronized Network Timing source is required, it is possible to propagate clock to downstream devices using a T1/E1 port on a TAC card on the MXK.

To view current source of clocking on the MXK, enter **clkmgrshow**. Timing is set locally on the uplink card or the TAC card.

In this case, the timing on the MXK is from the uplink card.

```
zSH> clkmgrshow
All lines are using LOCAL clock
```

In this case, the timing on the MXK is from the TAC card.

```
zSH> clkmgrshow
Primary system clock is 1/14/1/0 : T1
Secondary system clock is LOCAL timing
```

To view all the devices on the MXK capable of providing timing to the system, enter **clkmgrshow list**.

In this case, local timing from the uplink card is available on this MXK.

```
zSH> clkmgrshow list
eligible list has 0 entries
ineligible list has 0 entries
pending list has 0 entries
```

In this case, an TAC card is available to provide system timing on this MXK.

```

zSH> clkmgrshow list
eligible list has 1 entry
  1 *      eligible 1/14/1/0 ( 5) :      T1 : ACTIVE :      LOOP
ineligible list has 0 entries
pending list has 0 entries

```

Set the TAC card as clocking source

In cases where a synchronized Network Timing source is required, it is possible to propagate clock to downstream devices using a T1/E1 port on a TAC card on the MXK.



Note: The TAC card type for Europe is *ts1*.

Configuring a TAC card to be the primary timing source

- 1 Enter **slots** to view available TAC cards.

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  1: MXK 20 ACT ETH (RUNNING)
  10: MXK 8 PORT GPON (RUNNING)
  11: MXK 4 PORT GPON (RUNNING)
  14:*TAC ITM RING (RUNNING)

```

- 2 Change the **transmit-clock-source** parameter from *throughtiming* to *looptiming*.

```

zSH> update ds1-profile 1-14-1-0/ds1
ds1-profile 1-14-1-0/ds1
Please provide the following: [q]uit.
line-type: -----> {esf}:
line-code: -----> {b8zs}:
send-code: -----> {sendnocode}:
circuit-id: -----> {ds1}:
loopback-config: -----> {nolop}:
signal-mode: -----> {none}:
fdl: -----> {fdlnone}:
dsx-line-length: -----> {dsx0}:
line-status_change-trap-enable: --> {enabled}:
channelization: -----> {disabled}:
ds1-mode: -----> {csu}:
csu-line-length: -----> {csu00}:
clock-source-eligible: -----> {eligible}:
transmit-clock-source: -----> {throughtiming} looptiming
cell-scramble: -----> {true}:
coset-polynomial: -----> {true}:
protocol-emulation: -----> {network}:
signal-type: -----> {loopstart}:

```

```

ds1-group-number: -----> {0}:
line-power: -----> {disabled}:
timeslot-assignment: -----> {0}:
transmit-clock-recovery: -----> {none}:
transmit-clock-adaptive-quality: -> {stratum3}:

```

```

.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

```

zSH> update system-clock-profile 1-14-1-0/ds1

```

```

system-clock-profile 1-14-1-0/ds1
Please provide the following: [q]uit.
system-clock-eligibility: -> {false}: true
system-clock-weight: -----> {5}:

```

```

.....
Save changes? [s]ave, [c]hange or [q]uit: s

```

```

APR 16 14:00:43: warning: 1/a/1053: clkmgr: Secondary clock source set to 1/
14/1/0 Record updated.

```

```

zSH> APR 16 14:00:44: warning: 1/a/1053: clkmgr: System clock source set
to 1/14/1/0

```

```

APR 16 14:00:44: warning: 1/a/1053: clkmgr: There is no secondary clock

```

```

zSH> clkmgrshow

```

```

Primary system clock is 1/14/1/0 : T1
Secondary system clock is LOCAL timing

```

Setting clocking for the MXK from an TAC card is performed by configuring the **system-clock-profile** of the TAC card. For the **system-clock-weight** parameter, 1 is most preferred, 10 is least preferred.

```

zSH> show system-clock-profile
system-clock-eligibility:-> true false
system-clock-weight:-----> {1 - 10}

```

MXK alarms

This section describes the following:

- [Alarm manager, page100](#)
- [Alarm suppression, page101](#)

Alarm manager



Note: For GPON ONU alarms, refer to [GPON onu alarms on page71](#). The **alarm show** command does not display GPON ONU alarms.

The MXK central alarm manager includes the ability to view the active alarms on the system (using the **alarm** command) and the ability to store active alarms on the device. ZMS can use the alarms stored on the device to recreate the state of the alarms if it becomes disconnected.

The alarm command uses the following syntax:

alarm show [summary]

For example, the following command displays the number of current active alarms, the total number of alarms, the number of cleared alarms, as well as each active alarm and its severity:

```
zSH> alarm show
***** Central Alarm Manager *****
ActiveAlarmCurrentCount      :11
AlarmTotalCount              :36
ClearAlarmTotalCount         :25
OverflowAlarmTableCount      :0

ResourceId      AlarmType      AlarmSeverity
-----
1-a-2-0/eth     linkDown      critical
1-a-3-0/eth     linkDown      critical
1-a-6-0/eth     linkDown      critical
1-a-7-0/eth     linkDown      critical
1-a-8-0/eth     linkDown      critical
1-a-9-0/eth     linkDown      critical
1-a-10-0/eth    linkDown      critical
1-a-11-0/eth    linkDown      critical
1-2-2-1/other   linkDown      minor
system          power_supply_b_failure  warning
system          not_in_redundant_mode   major
```

The **summary** option displays the number of current active alarms, the total number of alarms, the number of cleared alarms:

```
zSH> alarm show summary
***** Central Alarm Manager *****
ActiveAlarmCurrentCount      :3
ActiveAlarmTotalCount        :3
ClearAlarmTotalCount         :0
OverflowAlarmTableCount      :0
```

Alarm suppression

The alarm suppression feature allows alarm/LED notification and output to be disabled based on alarm severity level for existing and future alarms. When an alarm level is disabled, all existing alarms of that type are cleared from the system. Future alarms of that type do not set LEDs or alarm relays and are not displayed in alarm output.

Alarm suppression is also supported in ZMS.

[Table11](#) lists the alarm suppression options and the resulting behaviors. By default, alarms for all severity levels are enabled.

Table 11: Alarm suppression options

Alarm Levels Enabled Setting	Alarm Behavior
critical+major+minor+warning	Enables all alarm levels. The default setting.
critical+major+minor	Disables all warning alarms.
critical+major	Disables all minor, and warning alarms.
critical+major+warning	Disables all minor alarms.
critical+minor+warning	Disables all major alarms.
critical+minor	Disables all major and warning alarms.
critical+warning	Disables all major and warning alarms.
critical	Disables all major, minor, and warning alarms.
major	Disables all critical, minor, and warning alarms.
major+minor+warning	Disables all critical alarms.
major+minor	Disables all critical and warning alarms.
major+warning	Disables all critical and minor alarms.
minor	Disables all critical, major, and warning alarms.
minor+warning	Disables all critical and major alarms.
(no levels)	Disables all alarm levels.

This example disables alarm/LED notification and output for all current and future alarms with the severity levels minor and warning.

```

zSH> update system 0
system 0
Please provide the following: [q]uit.
syscontact: -----> {Zhone Global Services and Support 7001 Oakport
Street Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113
support@zhone.com}:
sysname: -----> {Zhone MxK}:
syslocation: -----> {Oakland}:
enableauthtraps: -----> {disabled}:
setserialno: -----> {0}:
zmsexists: -----> {false}:
zmsconnectionstatus: --> {inactive}:
zmsipaddress: -----> {0.0.0.0}:
configsyncexists: -----> {false}:
configsyncoverflow: ---> {false}:
configsyncpriority: ---> {high}:
configsyncaction: -----> {noaction}:
configsyncfilename: ---> {}:

```

```

configsyncstatus: -----> {syncinitializing}:
configsyncuser: -----> {}:
configsyncpasswd: -----> {** private **}: ** read-only **
numshelves: -----> {1}:
shelvesarray: -----> {}:
numcards: -----> {3}:
ipaddress: -----> {0.0.0.0}:
alternateipaddress: ---> {0.0.0.0}:
countryregion: -----> {us}:
primaryclocksource: ---> {0/0/0/0/0}:
ringsource: -----> {internalringsourcelabel}:
revertiveclocksource: -> {true}:
voicebandwidthcheck: --> {false}:
alarm-levels-enabled: -> {critical+major+minor+warning}: critical+major
userauthmode: -----> {local}:
radiusauthindex: -----> {0}:
secure: -----> {disabled}:
webinterface: -----> {enabled}:
options: -----> {NONE(0)}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```


3 MXK IP SERVICE LEVEL AGREEMENT (IPSLA)

This chapter describes IP Service Level Agreement (IPSLA) on the MXK.

IPSLA configuration

The IP Service Level Agreement (IPSLA) feature assists service providers and network operators with enforcing and monitoring access network connections and performance. IPSLA uses ICMP Ping messages over configured IPSLA paths to track Round Trip Times (RTTs) and EHCO REQs/RSPs between initiator and responder devices to determine network performance and delays. Typically, one initiator device is used to monitor other responder devices in the network. A maximum of 32 IPSLA paths can be configured per MXK.

Initiator devices must be running IPSLA to request data for a responder device. Responder devices must be accessible through the **ping** command in the IP network, but do not need to run IPSLA. Responder devices not running IPSLA display limited statistical data and functionality. MXK can function as either an initiator or responder device.



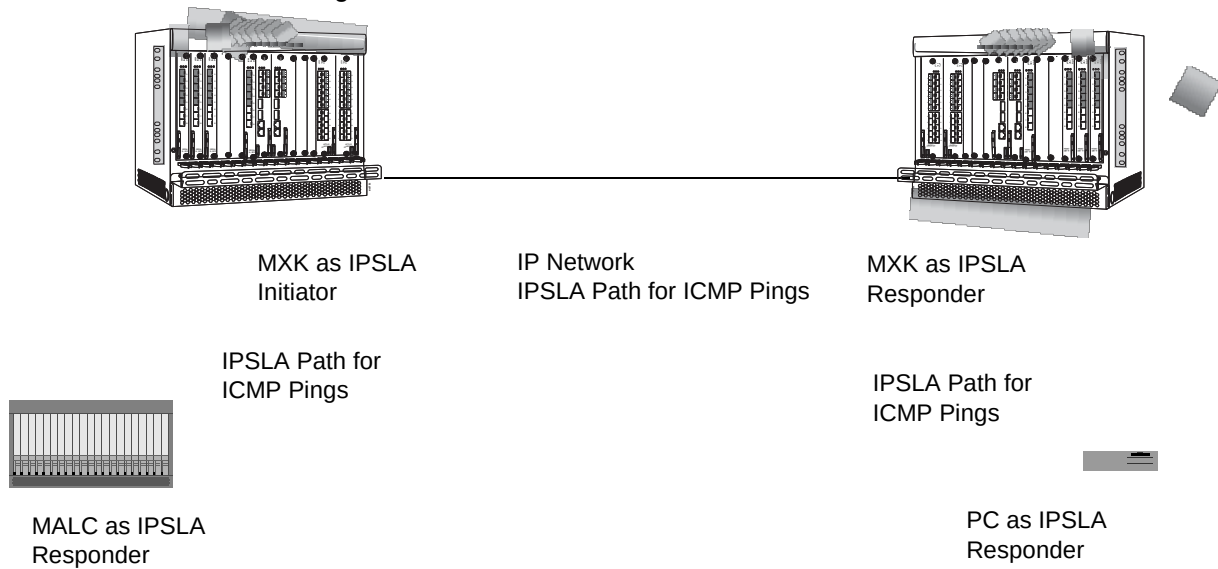
Note: Networks must support CoS queues and DSCP to provide valid per CoS statistics. Otherwise, all statistics are sent to the default CoS queue.

Default CoS-actions are assigned to each CoS queue so threshold crossing alarms can be configured to generate system alarms when thresholds are crossed for uptime, latency, jitter, and packet size.

Data based on received/sent packets and train rates is collected and displayed as real-time statistics for the current 15 minute interval as well as over 96 15-minute intervals for 24 hour historical statistics.

By default, IPSLA is disabled on all MXKs.

Figure 6: IPSLA



Configuring IPSLA

IPSLA requires the following configuration steps:

- Set ipsla-global settings to enable device state and optionally set polling interval
- Create ICMP path between devices
- Optionally, modify CoS actions for the desired CoS queues
- Optionally modify CoS map for Diff Server Control Point (DSCP) mappings

To configure IPSLA:

- 1 Display the global IPSLA settings and update the state and polling interval. The polling interval (60 to 3600 seconds) is used for real-time and historical statistics.

```
zSH> ipsla show global
state: -----> {disabled}
pollSeconds: -> {60}
```

Using the IPSLA command, enable IPSLA and set the polling interval to 120 seconds.

```
zSH> ipsla modify global state enabled pollseconds 120
```

- 2 Create a ICMP path between devices. The device on which this command is entered becomes the initiator device, while the device for which an IP address is entered becomes the responder device. Typically, one initiator device can be used to monitor other responder devices in the network for a maximum of 32 MXKs.



Note: Broadcast, multicast, and loopback addresses are not allowed.

```
zSH> ipsla add path 172.16.78.11
```

```
zSH> ipsla show path
```

```
Path configuration for ipAddress: 172.16.78.11
```

```
forwarding: -> {disabled}
```

```
state: -----> {enabled}
```

Modify the path using the IPSLA **modify path** command. This example disables the static path on device 192.168.254.17.

```
zSH> ipsla modify path ipaddress 192.168.254.17 state disabled
```

Delete a path using the IPSLA **delete** command.

```
zSH> ipsla delete path ipaddress 192.168.254.17
```



Note: Disabling or deleting the path or globally disabling the IPSLA feature will reset historical data.

- 3 Modify the default CoS actions to specify the response and threshold behavior for each CoS Action Index (1-8). These CoS actions map respectively to the CoS queues (0-7). [Table12](#) describes the CoS actions that are defined by default.

Table 12: CoS actions

Default Name	CoS Action Index	CoS Queue
Default	1	0
AFClass 1	2	1
AFClass 2	3	2
AFClass 3	4	3
AFClass4	5	4
Cos-5	6	5
ExpFwd	7	6
NetwCtrl	8	7

Table13 describes the parameters for each CoS action.

Table 13: CoS action parameters

Parameter	Description	Default
Name	Name of the IPSLA CoS action, up to 9 characters in length.	(1) Default, (2) AFClass1, (3) AFClass2, (4) AFClass3, (5) AFClass4, (6) Cos-5, (7) ExpFwd, (8) NetwCtrl.
Traps	Specifies whether a trap is issued when any SLA performance error threshold within this CoS is crossed.	Disabled
Timeouts	Specifies the number of consecutive missed IP SLA responses within this CoS before a zhoneIpSLATimeoutTrap is issued.	3 timeouts
Timeout Clear	Specifies the number of consecutive IPSLA responses within this CoS which must be received before the timeout error condition is cleared.	1 sample
Latency	Specifies the 15 sample average roundtrip latency value which must be exceeded within this CoS before a zhoneIpSLALatencyTrap is issued.	10000 milliseconds
Latency Clear	Specifies the number of consecutive IPSLA latency samples for which the 15 sample average roundtrip latency must be below the configured SLA latency error threshold within this CoS before the latency error condition is cleared.	1 sample
Jitter	Specifies the 15 sample roundtrip jitter value which must be exceeded within this CoS before a zhoneIpSLAJitterTrap is issued.	10000 milliseconds
Jitter Clear	Specifies the number of consecutive IPSLA RTT samples for which the 15 sample roundtrip jitter must be below the configured SLA jitter error threshold within this CoS before the jitter error condition is cleared.	1 sample
Packetsize	Specifies the minimum IPSLA Ping packet size in bytes. The range is 64 thru 2048 if the target IP device is running IPSLA, 64 thru 512 otherwise.	64 bytes

Display the settings for an individual CoS action.

```

zSH> ipsla show cos-action cosactionindex 1
Cos Action Configuration for cosActionIndex: 1:
name: -----> {Default}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
packetSize: -> {64}

```

Display the settings for all CoS actions (1-8).

```

zSH> ipsla show cos-action

```

Cos Action Configuration for cosActionIndex: 1:

```
name: -----> {Default}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
packetSize: -> {64}
```

Cos Action Configuration for cosActionIndex: 2:

```
name: -----> {AFClass1}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
packetSize: -> {64}
```

Cos Action Configuration for cosActionIndex: 3:

```
name: -----> {AFClass2}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
packetSize: -> {64}
```

Cos Action Configuration for cosActionIndex: 4:

```
name: -----> {AFClass3}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
packetSize: -> {64}
```

Cos Action Configuration for cosActionIndex: 5:

```
name: -----> {AFClass4}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
packetSize: -> {64}
```

Cos Action Configuration for cosActionIndex: 6:

```
name: -----> {Cos-5}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
packetSize: -> {64}
```

Cos Action Configuration for cosActionIndex: 7:

```
name: -----> {ExpFwd}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
```

```
packetSize: -> {64}
```

Cos Action Configuration for cosActionIndex: 8:

```
name: -----> {NetwCtrl}
traps: -----> {disabled}
timeOuts: ---> {3}
latency: ----> {10000}
jitter: -----> {10000}
packetSize: -> {64}
```

To modify a *cos-action*, specify the desired parameters to change in the command line. This example enables traps for cosActionIndex 1.

```
zSH> ipsla modify cos-action cosactionIndex 1 traps enabled
```

- 4 Configure the desired CoS maps to modify the default DSCP to CoS Action Index mappings. By default, DSCP are mapped to CoS Action Index entries based of RFC 2599. [Table14](#) shows the default mappings. A CoS Action Index of 0 indicates that the DSCP is not used.

Table 14: Default DSCP mappings

DSCP	CoS Action Index
1	8
11, 13, 15	7
19, 21, 23,	6
27, 29, 31	5
35, 37, 39	4
41	3
47	2
49, 57	1
2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 17, 18, 20, 22, 24, 25, 26, 28, 30, 32, 33, 34, 36, 38, 40, 42, 43, 44, 45, 46 48, 50, 51, 52, 53, 54, 55, 56, 58, 59, 60, 61, 62, 63, 64	0

Display the CoS map for an individual CoS action or for all CoS actions.

```
zSH> ipsla show cos-map
dscpIndex: 1      cosActionIndex: 1
dscpIndex: 2      cosActionIndex: 0
dscpIndex: 3      cosActionIndex: 0
dscpIndex: 4      cosActionIndex: 0
dscpIndex: 5      cosActionIndex: 0
dscpIndex: 6      cosActionIndex: 0
dscpIndex: 7      cosActionIndex: 0
dscpIndex: 8      cosActionIndex: 0
dscpIndex: 9      cosActionIndex: 0
dscpIndex: 10     cosActionIndex: 0
```

```

dscpIndex: 11    cosActionIndex: 2
dscpIndex: 12    cosActionIndex: 0
dscpIndex: 13    cosActionIndex: 2
dscpIndex: 14    cosActionIndex: 0
dscpIndex: 15    cosActionIndex: 2
dscpIndex: 16    cosActionIndex: 0
dscpIndex: 17    cosActionIndex: 0
dscpIndex: 18    cosActionIndex: 0
dscpIndex: 19    cosActionIndex: 3
Type A<CR> to print all, <CR> to continue, Q<CR> to stop:

```

Specify the desired index values in the command line to change the mapping of the DSCP index 1 to CoS queue 7. This example changes the mapping of DSCP index 1 to CoS queue 7.

```
zSH> ipsla modify cos-map dscpindex 1 cosactionindex 7
```

To clear a CoS map, specify the desired index values in the IPSLA command to delete the mapping of the DSCP index for the CoS queue. This example clears the mapping of DSCP index 1 and resets it to the CoS queue 0.

```
zSH> ipsla modify cos-map dscpindex 1 cosactionindex 0
```

- 5 Display real-time statistics for path or CoS queue. Real-time statistics represent minimum, maximum, average, and current values over the current 15 minute polling period based on data collected for each polling intervals. For example, if the polling interval is configured for 60 seconds, the real-time statistics display the data compiled from the latest 15 60-second polling intervals contained in the current polling period.



Note: RTT values of 0 (zero) indicate a lack of data, while sub-millisecond RTTs are reported as 1.

These statistics can be displayed individually or collectively for a specified IP address or for all configured paths.

```
zSH> ipsla stats path ipaddress 192.168.254.15
```

```
zSH> ipsla stats path
```



Note: Current and historical statistics on redundant uplinks are not supported. On uplink protection switching, these statistics are reset to 0.

Table15 describes the statistics for the configured paths.

Table 15: Statistics for the configured IPSLA paths

Path Statistic	Description
Target IP Address	IP Address of the device which is at the other end of the path.

Table 15: Statistics for the configured IPSLA paths (Continued)

Path Statistic	Description
Target Name	Name of the remote device.
Target Type	Type of the remote device.
ACT	Availability status of the remote device.
Source IP	IP Address of the discovery source device.
CNX	Type of path either static or dynamic.
UpTime (sec)	Amount of time in seconds that elapsed since the last transition from Inactive to Active.
I/R	Role played by the local device in collection of latency and availability statistics. Initiator - Device that initiates the IPSLA ping packet used for statistics collection; Responder - Device that returns the IPSLA ping packet sent by the Initiator.
CoS Mismatch	Number of IPSLA ping packets received which indicate a mismatch between the Class Of Service (CoS) definitions at the remote unit and those of the source unit.

Display real-time CoS statistics individually or collectively by CoS action index, IP address or all CoS actions.

```
zSH> ipsla stats cos cosactionindex 1
```

```
zSH> ipsla stats cos ipaddress 10.2.1.254
```

```
zSH> ipsla stats cos
```

[Table16](#) explains the CoS Action Index statistics.

Table 16: CoS Action Index statistics

CoS Action Index Statistic	Description
CoS Index	Index number of the CoS Action Index.
Target IP Address	IP Address of the device which is at the other end of the path.
Last RTT	RTT reported in the most recent successful ping attempt.
Min RTT	Smallest RTT since this statistic was last cleared to a zero value.
Avg RTT	Average RTT since this statistic was last cleared to a zero value.
Max RTT	Largest RTT since this statistic was last cleared to a zero value.
Drop Resp	Number of failed pings since this statistic was last cleared to a zero value.

Display historical statistics individually or collectively based on IP address, CoS action index, and index value of a 15 minute interval. Historical statistics are displayed for the latest 24 hour period or a specified 15 minute interval within the latest 24 hour period.

For historical statistics, IPSLA averages values for the most recent 96 15-minute intervals and displays the minimum, maximum, average and current values in a table for a 24 hour summary.

```
zSH> ipsla stats history cosactionindex 1
Up to 96 intervals....
```

```
zSH> ipsla stats history ipaddress 10.2.1.254
```

```
zSH> ipsla stats history index 1
```

```
zSH> ipsla stats history
Up to 96 intervals....
```

Each bulk statistic relies on a bulk-statistics profile to define the OID, instance and other MIB information used to collect and display the data. When an IPSLA path is modified or deleted during the process of data collection, the related bulk-statistics profiles may lose their association and become dangling profiles.

The **bulkstats audit** command enables users to check for and delete dangling bulk-statistics profiles. The bulkstats audit command provides an interactive and repair option. The interactive option lists all dangling profiles with the option to modify or delete the profile. The repair option prompts for profile deletion.

bulkstats audit -interactive | repair

To display and repair dangling bulk-statistics profiles, enter the **bulkstats audit** command.

```
zSH> bulkstats audit -interactive
Checking validity.....
3 dangling profiles found.

bulk-statistic 5
enabled: -----> {true}
oid: -----> {zhoneIpSLAPathStatByCOSAvgRTT}
instance: -----> {6.1.11.1.15.253}
include-children: -> {false}

[d]elete, [m]odify, [n]ext, [p]revious, [h]elp, [q]uit ? d

bulk-statistic 55
enabled: -----> {true}
oid: -----> {zhoneIpSLAPathStatByCOSAvgRTT}
instance: -----> {2.1.173.24.95.2}
include-children: -> {false}

[d]elete, [m]odify, [n]ext, [p]revious, [h]elp, [q]uit ? d
```

```
bulk-statistic 555
enabled: -----> {true}
oid: -----> {zhoneIpSLAPathStatByCOSEvgRTT}
instance: -----> {2.1.173.24.72.103}
include-children: -> {false}
[d]elete, [m]odify, [n]ext, [p]revious, [h]elp, [q]uit  d
```

```
zSH> bulkstats audit -repair
Checking validity.....
1 dangling profile found.
Delete profile? { [y]es or [n]o }  y
```

4 IP CONFIGURATION

This chapter explains Internet Protocol (IP) services on the MXK. It includes the following sections:

- [IP overview, page115](#)
- [Routing overview, page123](#)
- [DHCP overview, page126](#)
- [MXK routing configurations, page132](#)
- [IP administrative procedures, page171](#)
- [CPE Manager, page181](#)

Other IP related information can be found in:

- [Chapter 6, Video Configuration](#)

IP overview

This section describes:

- [IP services, page116](#)
- [IP protocols, page116](#)
- [IP Quality of Service \(QoS\), page118](#)
- [Floating IP interfaces, page123](#)

The Internet Protocol (IP) is a network-layer (Layer 3) protocol that contains addressing information and some control information that enables packets to be routed. IP is documented in RFC 791 and is the primary network-layer protocol in the Internet protocol suite.

The MXK supports both routing (Layer 3) and bridging (Layer 2) configuration. Layer 2, the lower level, also called the logical link layer, uses Media Access Control (MAC) addresses to direct traffic and layer 3 uses IP addresses to direct traffic.

Since both routing and bridging are created on logical interfaces associated with physical ports, the same physical port can support a logical interface configured for routing, and a logical interface configured for bridging. When configuring the MXK for bridging and routing, separate VLANs must be used.

IP services

The MXK provides the following IP features:

- IP forwarding and routing — incoming packets from an interface are forwarded to the appropriate output interface using the routing table rules.
- Local DHCP server.
- DHCP relay to provide access to upstream DHCP servers.
- Numbered IP interfaces and host-based IP interfaces.
- IP Type of Service (ToS).
- IP redundancy.

IP protocols

The MXK supports these IP protocols:

DNS

Domain Name System (DNS) maps domain names to IP addresses, enabling the system to reach destinations when it knows only the domain name of the destination.

DHCP

Dynamic Host Control Protocol (DHCP) is the means for dynamically assigning IP addresses. Basically, a DHCP server has a pool of IP addresses that can be assigned to DHCP clients. A DHCP client maintains its MAC address, but may have a different IP address each time it connects to the network. DHCP simplifies network administration since the DHCP server software tracks the used and unused IP addresses.

DHCP also provides a mechanism through which clients obtain configuration parameters such as the default router, the DNS server, subnet mask, gateway address, lease time, and IP address from the DHCP server.

When the MXK acts as a local DHCP server, the MXK can assign temporary (leased) IP addresses to clients. Each DHCP client sends a request to the MXK for an IP address lease. The MXK then assigns an IP address and lease time to the client. The MXK keeps track of a range of assignable IP addresses from a subnetwork.

Some customers choose to have the same IP address every time their DHCP lease renews. This is known as sticky IP addresses. By default, the MXK attempts to assign the same IP address to the same client on DHCP lease renewal.

With shared DHCP pools (or subnet groups), DHCP servers are not linked to physical interfaces providing routing configurations a range of addresses. It is

also possible for Zhone devices to assign blocks of IP addresses specifically for certain customers.

The MXK may also act as a DHCP relay agent, supporting DHCP requests from downstream devices to upstream DHCP servers. The MXK supports primary and alternate DHCP server configurations. DHCP relay supports Option 82 insertion to identify the requesting client to the DHCP server.

Routing Information Protocol (RIP)

Routing Information Protocol (RIP), an interior gateway protocol (IGP), is widely used for routing traffic on the Internet. RIP performs routing within a single autonomous system. It is based on distance-vector algorithms that measure the shortest path between two points on a network. The shortest path is determined by the number of hops between those points. RIP routers maintain only the best route (the route with the lowest metric value) to a destination. After updating its routing table, the router immediately begins transmitting routing updates to inform other network routers of the shortest route. The MXK supports RIPv1 and RIPv2.

IP Quality of Service (QoS)

This section discusses MXK QoS features including:

- [ToS, CoS, and sCoS on IP interfaces, page118](#)
- [ToS fields in an IP header, page119](#)
- [CoS fields in a VLAN header, page119](#)
- [ToS, CoS, sCoS parameters, page120](#)
- [802.1p priority queues, page122](#)

ToS, CoS, and sCoS on IP interfaces

The MXK supports the marking and remarking of ToS values in IP packets and Class of Service (CoS) values in Ethernet VLAN headers as defined by IETF RFC1349 and IEEE 802.1p respectively. The configured ToS and CoS levels specify packet priority and queueing used to transport the packet through the Ethernet and IP networks. The MXK sets and transports the ToS/CoS values, while the switches and routers connected to the MXK perform the QoS priority and queueing services.

CoS is set in tagged or s-tagged (VLANs or SLANs) on the outgoing interface and ToS is set in untagged (no VLANs or sLANs) configurations also on the outgoing interface. Both CoS and ToS can be set in the same outgoing interface to ensure that important services like voice and video retain packet priority and queueing throughout the network. As packets travel through the network, routers will recognize ToS values and switches will recognize CoS values.



Note: When configuring the MXK for voice, a separate ToS value for VoIP signalling packets must be set in the **voip-server-entry** profile.

See [Advanced VoIP configuration on page587](#).

The MXK QoS features allow you to:

- Add IP packet ToS values and VLAN header CoS and sCoS values to packets originating from the MXK.
- Add IP packet ToS values to voice signalling packets by configuring the **ipTos** parameter in the **voip-server-entry** profile.
- Overwrite existing IP packet ToS values that are transported through the MXK network facing IP interface when the **tosOption** parameter in the **ip-interface-record** profile is set to *all* by inserting the ToS value found in the network facing IP interface.
- Leave existing IP packet ToS values unchanged as they pass through the MXK network facing IP interface when the **tosOption** in the **ip-interface-record** profile is set to *originate*.

ToS fields in an IP header

IP packets have a ToS byte in their headers that contains information about relative priority. The ToS byte is divided into two fields called IP Precedence and ToS. The IP Precedence field contains a 3-bit priority designation. Most normal traffic has an IP Precedence value of zero. Higher values in this field indicate that traffic is more important and that it requires special treatment. IP Precedence values greater than 5 are reserved for network functions.

The ToS field indicates the queueing priority value based on eight (0-7) levels of service. This field contains information about how the traffic should be forwarded. The MXK supports basic ToS marking without queue servicing options in the **ip-interface-record** profile. Packets marked based on a configurable profile to let the system know which bits use which queue.



Note: When setting ToS for IP packets in the **ip-interface-record** profile, the values in the precedence bits column are used.

When configuring the MXK for voice, a separate ToS value for VoIP signalling packets must be set in the **voip-server-entry**.

See [Advanced VoIP configuration on page 587](#).

Table 17 specifies the IP ToS settings used in the **voip-server-entry** profile based on IP Precedence bits.

Table 17: IP ToS settings and IP Precedence bits

Precedence bits	ToS value
0 (Routine)	0
1 (Priority)	32
2 (Immediate)	64
3 (Flash)	96
4 (Flash override)	128
5 (CRITIC/ECP.)	160
6 (Internetwork control)	192
7 (Network control)	224

CoS fields in a VLAN header

The VLAN header in Ethernet packets contains a CoS field for queueing priority or Class of Service (CoS) values based on eight (0-7) levels of service. This field contains information about how the traffic should be forwarded. The MXK supports basic CoS marking and remarking without any queue servicing options. Packets marked or remarked based on a configurable profile to let the system know which bits use which queue.

CoS values range from 0 — 7, with the lowest priority being 0 and the highest priority 7. The MXK supports eight queues per physical interface as follows — frames with a 0 CoS value is put into queue number 0; frames with a 1 CoS value is put into queue number 1; and so forth.

These are strict priority queues which mean that everything is cleared out of the high priority queue first. Only after that queue is empty is the next queue serviced. Since these are strict priority queues it is possible that the lower priority queues may get overloaded while the higher priority queues are being cleared.

Frames which require the highest throughput or are sensitive to latency (the amount of time between received packets) should be in higher priority queues. Since queuing is relative to the type of traffic, the priority settings depend on the type of traffic. Normally video and voice are more sensitive to throughput and latency issues.

ToS, CoS, sCoS parameters

[Table18](#) describes the parameters in the **ip-interface-record** profile used for ToS and CoS support.

For more information on how to configure IP packets for CoS and ToS and how to configure voice signalling packets for ToS, see [Advanced VoIP configuration on page587](#).

Table 18: ip-interface-record profile ToS and CoS parameters

Parameter	Description
tosOption	Specifies how to handle the IP ToS precedence and VLAN header CoS bits. Values: Disable Leave any existing ToS and CoS values unchanged. The default setting. Originate Replace the current ToS and CoS values in all packets originating from the current device. ToS and CoS values in packets that are transported through (not originating on) this MXK are not affected. The ToS value is specified in the tosCos field. The CoS value is specified in the vlanCOS field. All Replace the current ToS and CoS values in all packets originating and transported through this device. The ToS value is specified in the tosCos field. The CoS value is specified in the vlanCOS field. This setting has no affect on VoIP RTP packets originated from this interface.
tosCOS	Specifies the value loaded into the ToS precedence bits in the IP header for packets originating and transported through the current device. Value range is 0 to 7. Default is 0.

Table 18: ip interface record profile ToS and CoS parameters (Continued)

Parameter	Description
vlanCOS	Specifies the value loaded into the CoS field of the VLAN header for packets originating and transported through the current device. Value range is 0 to 7. Default is 0.
s-tagIdCOS	Specifies the value loaded into the sCoS field of the SLAN header for packets originating and transported through the current device. Value range is 0 to 7. Default is 0. If present, this outer tag controls the behavior.

To view the ToS and CoS settings in the **ip-interface-record** profile, enter **show ip-interface-record**.

```

zSH> show ip-interface-record
vpi:-----> {0 - 4095}
vci:-----> {0 - 65535}
rdindex:-----> {0 - 2147483647}
dhcp:-----> none client server both
addr:-----> {0 - -1}
netmask:-----> {0 - -1}
bcastaddr:-----> {0 - -1}
destaddr:-----> {0 - -1}
farendaddr:-----> {0 - -1}
mru:-----> {0 - 2147483647}
reasmmmaxsize:-----> {0 - 65535}
ingressfiltername:-----> {33}
egressfiltername:-----> {33}
pointtopoint:-----> no yes
mcastenabled:-----> no yes
ipfwdenabled:-----> no yes
mcastfwdenabled:-----> no yes
natenabled:-----> no yes
bcastenabled:-----> no yes
ingressPacketRuleGroupIndex:--> {0 - 2147483647}
egressPacketRuleGroupIndex:--> {0 - 2147483647}
ipaddrdynamic:-----> static ppp dhcpclient unnumbered cpemgr
dhcpserverenable:-----> true false
subnetgroup:-----> {0 - 2147483647}
unnumberedindex:-----> {0 - 2147483647}
mcastcontrollist:-----> {264}
vlanid:-----> {0 - 4090}
maxVideoStreams:-----> {0 - 210}
tosOption:-----> disable originate all
tosCOS:-----> {0 - 7}
vlanCOS:-----> {0 - 7}
s-tagTPID:-----> {33024 - 37376}
s-tagId:-----> {0 - 4090}
s-tagIdCOS:-----> {0 - 7}

```

[Table19](#) describes the ToS parameter in the **voip-server-entry** profile used for ToS support.

Table 19: voip-server-entry ToS parameter

Parameter	Description
ipTos	This parameter indicates the tos value that is set in the IP header for voice IP traffic.

The **ipTos** parameter in the **voip-server-entry** sets the priority for voice traffic.

To view the ToS settings available in the **voip-server-entry** profile, enter **show voip-server-entry**.

```

zSH> show voip-server-entry
zhoneVoipServerAddrType:----->  unknown  ipv4  ipv6  ipv4z  ipv6z  dns
zhoneVoipServerAddr:----->      {255}
zhoneVoipServerUdpPortNumber:----->  {1 - 65535}
zhoneVoipServerId:----->          longboard asterisk sipexpressrouter
metaswitch sylantrio broadsoft ubiquity generalbandwidth tekelec-t6000 generic
sonus siemens tekelec-t9000 lucent-telica nortel-cs2000 nuera lucent-imerge
coppercom newcross cisco-bts cirpack-utp italtel-issw cisco-pgw microtrol-msk10
nortel-dms10 verso-clarent-c5cm cedarpoint-safari huawei-softx3000 nortel-cs1500
taqua-t7000 utstarcom-mswitch broadsoft-broadworks broadsoft-m6 genband-g9
netcentrex genband-g6
protocol:----->                  sip  mgcp  megaco  ncs
sendCallProceedingTone:----->    true  false
rtcpEnabled:----->               true  false
rtcpPacketInterval:----->       {0 - 0}
interdigitTimeout:----->        {0 - 0}
ipTos:----->                    {0 - 255}
systemDomainName:----->         {260}
expires-invite-value:----->     {0 - 2147483647}
expires-register-value:----->   {0 - 2147483647}
expires-header-method:----->    invite+register
session-timer:----->             on  off
session-expiration:----->       {90 - 2147483647}
session-min-session-expiration:---> {90 - 2147483647}
session-caller-request-timer:-----> yes  no
session-callee-request-timer:-----> yes  no
session-caller-specify-refresher:-> uac  uas  omit
session-callee-specify-refresher:-> uac  uas
dtmf-mode:----->                inband  rfc2833
rtp-termid-syntax:----->        {96}

```

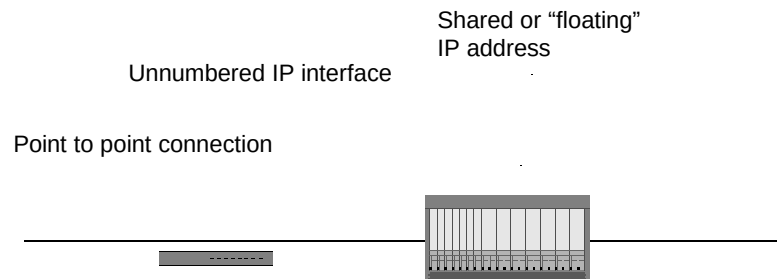
802.1p priority queues

Multi-media Traffic Management (MTM), is a rules-based policy enforcement mechanism for SLMS systems. The MXK MTM is used to mark packet priorities and service queues. The MXK supports eight strict priority queues on each port. The scheduling policy is "strict priority", where the higher priority queues are serviced until empty as part of the MXK's implementation of the MTM feature set for QoS.

Floating IP interfaces

Floating IP interfaces reduce the number of IP addresses used by a device. Floating interfaces are like other point-to-point connections, except a “floating” or virtual IP interface is used as the local IP address in the **ip-interface-record**. See [MXK routing configurations on page132](#) for procedures using floating IP interfaces.

Figure 7: Unnumbered IP interfaces



Routing overview

This section discusses:

- [Host-based \(unnumbered\) routing overview, page124](#)
- [Network-based \(numbered\) routing overview, page125](#)

Routing is the process of selecting a next hop for forwarding data traffic based on IP address. The routing information base (RIB) contains all the information about the routes in the system, including the preference values and interface states. The forwarding information base (FIB) is derived from the RIB and only contains the best route to a given destination.

IP routing through the system makes use of the following types of routes:

- Interface routes—These routes are defined by the addresses and netmasks that are provisioned on the IP interfaces.
- Static routes—Routing defines the paths over which packets travel in the network. Static routes are manually configured for a section of the network and are used in place of dynamic routing. A static route defines the path in terms of an interface identifier or the IP address of a next-hop router on a directly attached network.

There are two kinds of static routes:

- Low preference—These routes are only used to define default routes (that is, routes of last resort) and are less preferable to most other routes.

- Normal preference—All other static routes are considered more preferable than other types of routes (with the exception of interface routes).
- Dynamic routes—These routes are learned by running routing protocols, such as RIP, and have varying preferences, depending on how they were learned.

[Table 20](#) describes the default routing preferences on the device. These preferences cannot be overridden. Higher numbers indicate more preferred route types:

Table 20: Routing preferences

Type of route	Default preference
Local	10
Static	9
RIP	4
Static low (used for default routes)	4

Host-based (unnumbered) routing overview

Host-based routing uses either static IP configuration or dynamic IP configuration with a floating IP interface to add a single IP address to the routing table for each route. This type of routing allows a granular allocation of addresses based on the host floating IP address and the available subnetwork addresses. Configure host-based routing with the **host add** command. Each host is configured with a reference to a floating IP interface so that when an IP address is added to the routing table for the host, the address is assigned to the host from the floating IP subnet.

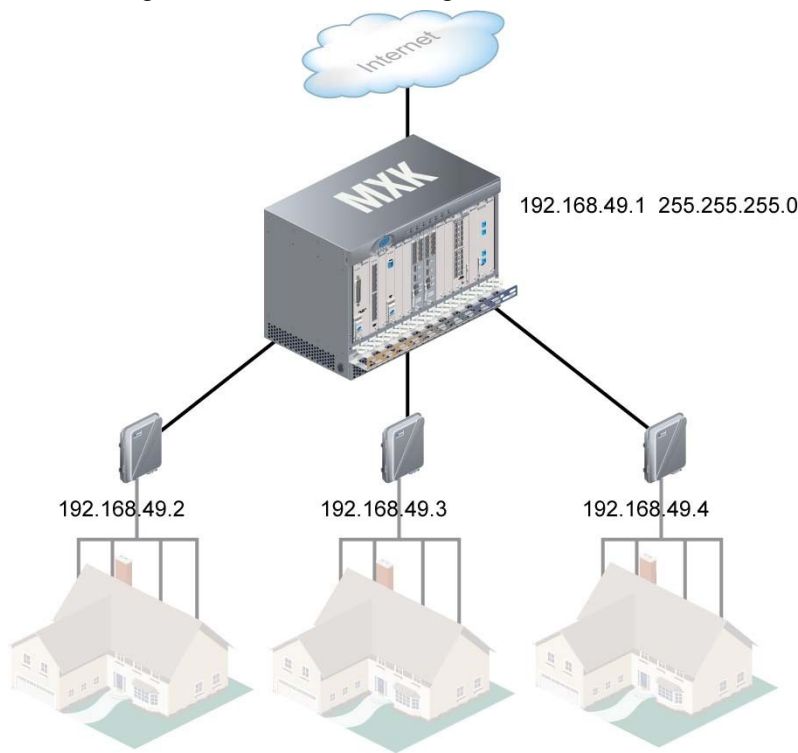
For example, an floating host address of 10.10.10.1/24, adds one entry in the routing table for the address 10.10.10.1 and makes available a subnet of 253 addresses for individual route configuration. When a route is added to a host, a new routing table entry is created.

The **host add** command can also assign VLAN, SLAN, CoS, and sCoS values to the host interface. In the **host add**, **host modify** and **host delete** commands, <slot> and <port> may be replaced with brackets containing numbers in series and/or (dash-separated) ranges; <port> may be replaced with wildcard '*' for all ports on the card.

Host-based routing uses floating IP interfaces and shared DHCP pools to conserve IP addresses or a static IP address. In host-based routing, subscribers connected to the MXK are on the same subnet as the MXK floating interface.

See [Host-based routing on page 132](#) for host-based routing configuration procedures.

Figure 8: Host-based routing



Network-based (numbered) routing overview

Network-based routes are configured with the **interface add** command to create a numbered IP interface that adds IP network addresses with variable length subnet masks to the routing table. This type of routing allows a single routing table entry to represent many numbered host addresses. However, it does not allow for granular IP address allocation. For example, an interface configured with 10.10.10.1/24 adds just one entry to the routing table for 10.10.10.1/24. All 254 addresses in this subnet are assigned to this interface, regardless of how many addresses in this subnet are actually used.

Network-based routing adds large numbers of IP addresses. Unlike host-based routing, network based-routing requires numbered IP interfaces on the MXK and does not use floating IP addresses. In network-based routing, each host, connected to an interface, is in the same network as the MXK numbered interface.

See [Network-based routing on page152](#) for network-based configuration procedures.

Figure 9: Network-based routing

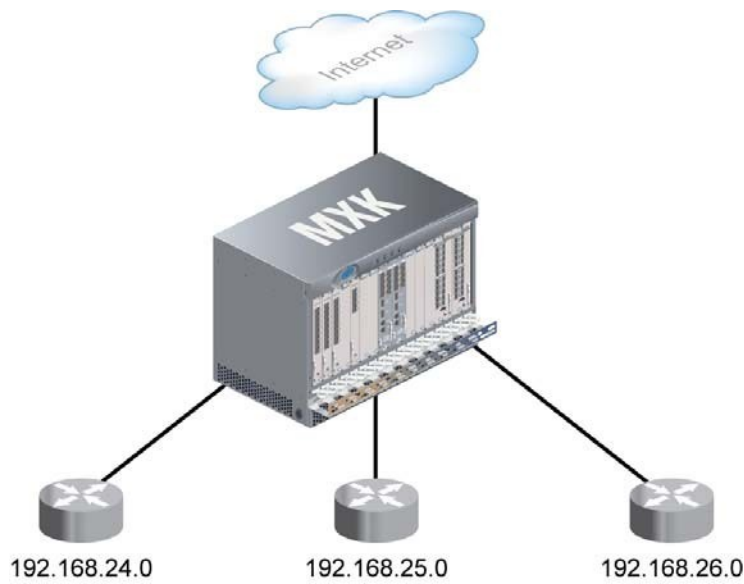


Table 21: Host- based and network-based commands for adding IP interfaces

Command	Application	IP Assignment	Address Allocation	Interface Type
Host add	Host-based routing with bridge or router	Static/Dynamic Server	Single per host add command	unnumbered
Interface add	Network-based routing with bridge or router	Static Server or client	Multiple based on subnet mask length	numbered

DHCP overview

This section discusses:

- [MXK DHCP server support, page126](#)
- [DHCP server profiles and scope, page127](#)
- [DHCP server options, page127](#)
- [DHCP server subnet options, page128](#)
- [MXK DHCP relay, page130](#)

The MXK uses Dynamic Host Control Protocol (DHCP) to dynamically assign IP address to physical devices. This conserves the number of IP addresses a network requires.

MXK DHCP server support

The MXK DHCP supports the following types of DHCP configurations:

- Dynamic address allocation, where the server chooses and allocates an IP address with a finite lease. By default, the MXK will attempt to assign the same address (if available) to a device on lease renewal. This default can be changed to force a new address to be assigned.
- Static address allocation, where the server allocates the same IP address every time a device connects to the network.

DHCP server profiles and scope

The MXK uses the following profiles to configure DHCP servers:

- **dhcp-server-subnet**—Defines options for a specific network that is being managed by the DHCP server. Settings in the **dhcp-server-subnet** record override the default address pool set up by the **dhcp-server-options** record. See [DHCP server subnet options on page128](#) for more information.
- **dhcp-server-options**—Configures a default profile that is used to generate default configurations for networks that are not explicitly configured. See [DHCP server options on page127](#) for more information.
- **ip-interface-record**—enables local DHCP on the MXK. The IP address defined in the **ip-interface-record** is used to determine the DHCP address pool for the attached network. See [Host-based routing on page132](#) and [Network-based routing on page152](#) for more information on configuring DHCP services.

The DHCP server looks for configuration settings in order from the most specific record, the **dhcp-server-host**, to the most general the **dhcp-server-options** record. It uses parameter settings in the following order:

1. **dhcp-server-host**
2. **dhcp-server-group**
3. **dhcp-server-subnet**
4. **dhcp-server-options**

If a parameter is set in multiple profiles (for example, lease times or default routers), the MXK uses the settings that are in the most specific record. This means that the DHCP server could use parameter settings in multiple records (if, for example, all client lease times were set in the **dhcp-server-options** record, and address ranges were set in the **dhcp-server-subnet** records.)

If only the **dhcp-server-options** record exists, the MXK uses those settings as the default for all DHCP server interfaces. For information about logging DHCP requests, see [DHCP logging on page175](#).

DHCP server options

At startup, the MXK creates a default **dhcp-server-options** record. This profile defines global options for configurations enabling DHCP.

The following shows the **dhcp-server-options** profile with its default values:

```
zSH> get dhcp-server-options 0
Please provide the following: [q]uit.
lease-time: -----> {43200}:
min-lease-time: -> {0}:
max-lease-time: -> {86400}:
reserve-start: --> {5}:
reserve-end: ----> {5}:
restart: -----> {no}:
```

[Table 22](#) describes the **dhcp-server-options** profile that supports the following configurable parameters (all others should be left at their default values):

Table 22: dhcp-server-options profile configurable parameters

Parameter	Description
lease-time	The global default time in seconds that will be assigned to a DHCP lease if the client requesting the lease does not request a specific expiration time.
min-lease-time	The minimum expiration time in seconds that will be assigned to a DHCP lease by the server, regardless of the value specified by a client. Values: 0 to 2147483647 Default: 0
max-lease-time	The maximum time in seconds that will be assigned to a lease regardless of the value specified by a client. Values: 0 to 2147483647 Default: 86400
reserve-start	The default number of IP addresses, at the beginning of the MXK subnet IP address space, that are reserved by the DHCP server. To override this default, create a specific subnet rule for each subnet that needs to be handled differently. Note: Be sure the subnet is large enough.
reserve-end	The default number of IP addresses at the end of the MXK 's subnet IP address space that are reserved by the DHCP server. To override this default, create a specific subnet rule for each subnet that needs to be handled differently. Note: Be sure the subnet is large enough.

DHCP server subnet options

The **dhcp-server-subnet** profile allows you to edit the options for a specific network that is being managed by the DHCP server. All subnets within a

routing domain must be unique, so a given subnet object will provide options for exactly one connected network.

[Table 23](#) describes the **dhcp-server-subnet** profile that supports the following configurable parameters (all others should be left at their default values):

Table 23: dhcp-server-subnet profile configurable parameters

Parameter	Description
network	The IP network address of this subnet.
netmask	The subnet mask associated with the IP interface. The value of the mask is an IP address with all the network bits set to 1 and all the hosts bits set to 0.
domain	The routing domain to which this subnet, group, or host parameter applies.
range1-start, range2-start, range3-start, range4-start	The starting IP address of an address pool in this subnet. If either the start or end range has a value of 0 then the entire address pool is ignored.
range1-end, range2-end, range3-end, range4-end	The ending IP address of an address pool in this subnet. If either the start or end range has a value of 0, then the entire address pool is ignored.
default-lease-time	The default time, in seconds assigned to a lease if the client requesting the lease does not request a specific expiration time. default: -1 The values of the DHCP server options profile are used.
min-lease-time	See description in dhcp-server-options profile.
max-lease-time	See description in dhcp-server-options profile.
boot-server	The IP address of the server from which the initial boot file (specified in the bootfile parameter) is to be loaded.
bootfile	The name of the initial boot file loaded by the client. The filename should be recognizable to the file transfer protocol that the client will be using to load the file if you have devices requiring bootp. If the device only needs IP addresses, this file is not needed.
default-router	The IP address of the client default gateway.
primary-name-server	The IP address of the primary domain name server that the client should use for DNS resolution.
secondary-name-server	The IP address of the secondary domain name server that the client should use for DNS resolution.
domain-name	The name of the DNS domain.
subnetgroup	A number which indicates which DHCP subnet group this pool is a member of. A value of 0 (default) indicates that the subnet is not a member of any group. Values specific to the subnet are set here.

Table 22: dhcp-server-subnet-profile-configurable-parameters (Continued)

Parameter	Description
stickyaddr	The DHCP server attempts to assign the same IP address to the same host, if possible, based on hardware address. Values: disable enable Default: enable
external-server	Enable a primary external subnet server in order to support DHCP relay agent. 0.0.0.0
external-server-alt	Enable an alternate external subnet server in order to support DHCP relay agent. Default: 0.0.0.0

```

zSH> get dhcp-server-subnet1
network:-----> {10.107.8.0} references floating IP address
netmask:-----> {255.255.255.0} the floating IP address range
domain:-----> {0}
range1-start:-----> {10.107.8.1} beginning floating IP address range
range1-end:-----> {10.107.8.250} ending floating IP address range
range2-start:-----> {0}
range2-end:-----> {0}
range3-start:-----> {0}
range3-end:-----> {0}
range4-start:-----> {0}
range4-end:-----> {0}
default-lease-time:----> {-1} lease times
min-lease-time:-----> {-1} lease times
max-lease-time:-----> {-1} lease times
boot-server:-----> {} for bootp services
bootfile:-----> {} for bootp services
default-router:-----> {10.107.8.254}
primary-name-server:---> {0.0.0.0} IP address for DNS server
secondary-name-server:-> {0.0.0.0} IP address for secondary DNS server
domain-name:-----> {} DNS domain name
subnetgroup:-----> {1} unique identifier for the subnet group
stickyaddr:-----> {enable}
external-server:-----> {0.0.0.0} external DHCP server IP address for DHCP relay only
external-server-alt:---> {0.0.0.0} alternate external DHCP server IP address for DHCP relay only

```

MXK DHCP relay

The **dhcp-relay** command enables you to add, modify, delete or show DHCP relay agents. The subnet address/mask will be derived from the system's floating IP address, if present, or may be specified NULL for use only with bridged interfaces. If multiple floating IP records are present, the desired <name>/<type> may be specified.

In DHCP relay configurations, the MXK serves as a DHCP relay agent that forwards DHCP discover and DHCP request packets to an external DHCP server. It then forwards the DHCP offer and DHCP ack/nak replies to the requesting DHCP host.

Broadcast messages are not allowed to go from device to device. The MXK can be configured as a DHCP relay agent that communicates with a DHCP server and acts as a proxy for DHCP broadcast messages that need to be routed to remote downstream devices.

Note the following requirements for DHCP relay:

- The external DHCP server must be configured to assign addresses on the same subnet as the floating IP.
- The external DHCP server must be configured with a static route for the remote device's subnet back to the MXK on which the relay agent is running. The DHCP server will send DHCP unicast packets to the relay agent's address.
- Different external servers can be used by different subnets.

MXK routing configurations

This section describes the following procedures:

- [Host-based routing, page132](#)
- [Network-based routing, page152](#)
- [Host-based routing for triple-play services on Ethernet, page158](#)
- [Host-based routing for triple-play services on GPON, page163](#)
- [Static routes, page168](#)
- [RIP configuration, page171](#)

Host-based routing

This section provides these configuration examples:

- [Static host-based routing without DHCP, page133](#)
- [Host-based routing with the MXK as a local DHCP server, page137](#)
- [Host-based routing with the MXK as a local DHCP server to provide DNS and bootp services, page140](#)
- [Host-based routing with an external DHCP server, page144](#)
- [Host-based routing with multiple dhcp-relay agents and one DHCP server, page147](#)
- [Host-based routing with an external DHCP server and an alternate DHCP server with dhcp-relay agent, page150](#)

Host-based routing uses a floating interface and adds a single IP address to the routing table for each route allowing a granular allocation of addresses based on the floating IP address and available subnet addresses.

Refer to the *CLI Reference Guide* for a complete description of the command options and syntax.

You can configure host-based routing on the MXK in one of three ways:

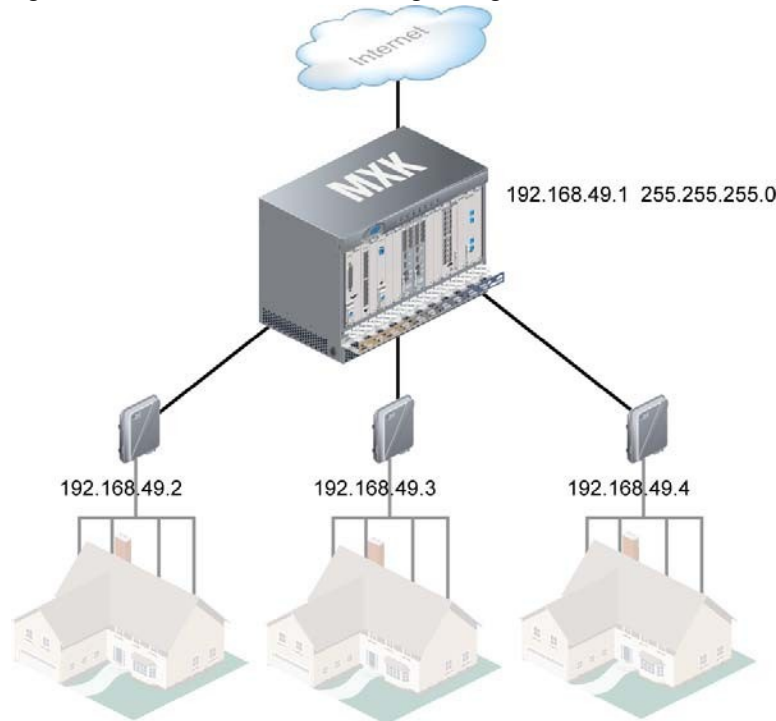
- Static configuration without a DHCP server.
See [Static host-based routing without DHCP on page133](#)
- DHCP services are on the MXK (the MXK is the DHCP server).
[Host-based routing with the MXK as a local DHCP server on page137](#)
and [Host-based routing with the MXK as a local DHCP server to provide DNS and bootp services on page140](#).
- The MXK uses an external DHCP server.

[Host-based routing with an external DHCP server on page144](#), [Host-based routing with multiple dhcp-relay agents and one DHCP server on page147](#), and [Host-based routing with an external DHCP server and an alternate DHCP server with dhcp-relay agent on page150](#).

Static host-based routing without DHCP

This procedure is for routing configurations statically without using DHCP, either locally or externally. [Figure10](#) displays a static host-based routing configuration.

Figure 10: Static host-based routing configuration



Configuring static host-based routing

- 1 Create a floating IP interface designating the IP address and subnet that will provide the IP addresses to all devices in the subnet.

```
zSH> interface add float pmt1 192.168.49.1 255.255.255.0
Created ip-interface-record pmt1/ip.
```

Verify the interface with the **list ip-interface-record interface/type** command.

For large configurations, simply entering **list ip-interface-type** may display more information than is useful.

```
zSH> list ip-interface-record pmt1/ip
ip-interface-record pmt1/ip
1 entry found.
```

2 View the **ip-interface-record** profile for *pmt1*.

View this interface to verify the range of the IP addresses available to assign to subscribers with the **host add** command.

The range of addresses provided by the *pmt1* interface is 192.168.49.2 — 192.168.49.254.

```
zSH> get ip-interface-record pmt1/ip
ip-interface-record pmt1/ip
vpi: -----> {0}
vci: -----> {0}
rdindex: -----> {1}
dhcp: -----> {none}
addr: -----> {192.168.49.1} floating IP address
netmask: -----> {255.255.255.0} net mask
bcastaddr: -----> {192.168.49.255} broadcast address for the subnet
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmxsize: -----> {0}
ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: --> {0}
egressPacketRuleGroupIndex: --> {0}
ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {false}
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {0}
maxVideoStreams: -----> {0}
tosOption: -----> {disable}
tosCOS: -----> {0}
vlanCOS: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}
```

3 Create a static IP interface for the host.

For static routing configurations without DHCP, each host is assigned an IP address from the range defined in the floating interface, in this case *pmt1*.

This example shows three IP routing interfaces created with static IP addresses.

```
zSH> host add 1-13-1-0/eth static 192.168.49.2
```

Adding host for 1-13-1-0/eth

```
zSH> host add 1-13-2-0/eth static 192.168.49.3
```

Adding host for 1-13-2-0/eth

```
zSH> host add 1-13-3-0/eth static 192.168.49.4
```

Adding host for 1-13-3-0/eth

4 Verify the host interface by entering **host show interface**.

For large configurations, simply entering **host show** may display unneeded amounts of data.

```
zSH> host show 1-13-1-0-eth
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.49.1	1-13-1-0-eth	0	S	192.168.49.2	

```
zSH> host show 1-13-2-0-eth
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.49.1	1-13-2-0-eth	0	S	192.168.49.3	

```
zSH> host show 1-13-3-0-eth
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.49.1	1-13-3-0-eth	0	S	192.168.49.4	

```
zSH> host show 1-13-3-0-eth
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.49.1	1-13-3-0-eth	0	S	192.168.49.4	

```
zSH> host show 1-13-3-0-eth
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.49.1	1-13-3-0-eth	0	S	192.168.49.4	

```
zSH> host show 1-13-3-0-eth
```

Deleting interfaces

1 Delete the static host IP interface.

```
zSH> host delete 1-13-1-0/eth ip 192.168.49.2
```

Deleting host for 1-13-1-0/eth

2 Delete the floating IP interface.

```
zSH> interface delete float pmt1
```

Interface pmt1 deleted

Delete hosts

There are several ways to use **host delete** to delete IP interfaces associated with an *interface/type*.

Deleting hosts using IP address

host delete <ip address> deletes the static host IP interface.

```
zSH> host delete 1-13-1-0/eth ip 192.168.49.2
```

Deleting host for 1-13-1-0/eth

Deleting hosts using unused

host delete unused <number> deletes the designated number of unassigned floating IP slots that have not yet been assigned an IP address.

```
zSH> host delete 1-13-2-0/eth unused 4
```

Deleting host for 1-13-2-0/eth

Deleting hosts using all

host delete all deletes *all* of the hosts on this subnet and the subnet itself.

```
zSH> host delete 1-13-1-0/eth all
Deleting host for 1-13-1-0/eth
```

Static and dynamic host configuration with the same subnet

The same subnet can be used for both static and dynamic configurations. Configuring dynamic hosts creates a **dhcp-server-subnet** profile where a range of addresses for static hosts can be reserved and a range of addresses for dynamic hosts can be defined as shown in [Figure11](#).

See [Host-based routing with the MXK as a local DHCP server on page137](#) for configuring a **dhcp-server-subnet** profile.

Figure 11: Example dhcp-server-subnet profile for static and dynamic addresses using the same subnet

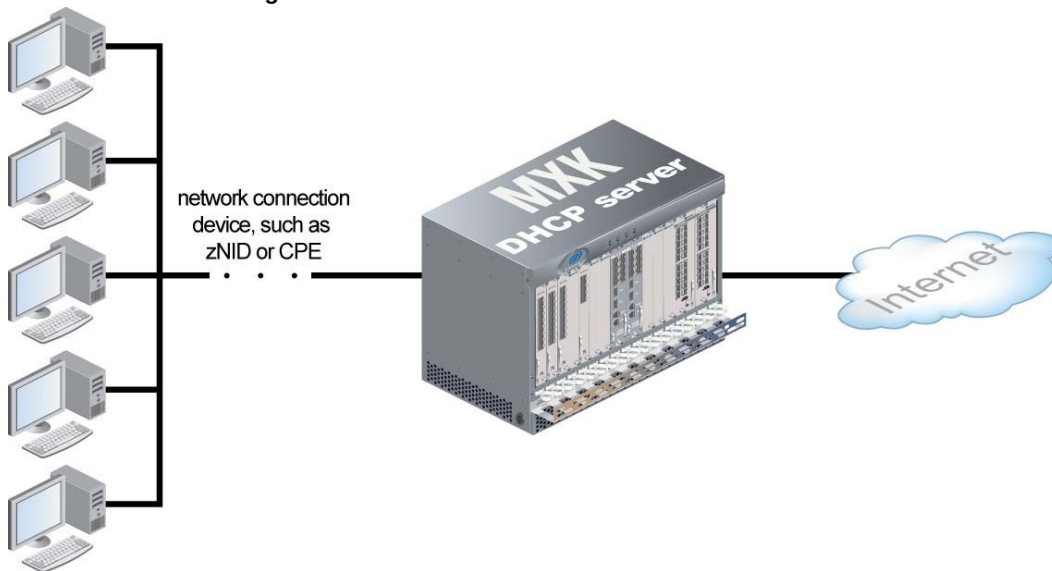
floating IP address	network: ----->	{10.107.8.0}:
subnet mask	netmask: ----->	{255.255.255.0}:
	domain: ----->	{0}:
reserve for static addresses	range1-start: ----->	{10.107.8.2}:
	range1-end: ----->	{10.107.8.25}:
defines range for dynamic addresses	range2-start: ----->	{10.107.8.26}:
	range2-end: ----->	{10.107.8.250}:
	range3-start: ----->	{0.0.0.0}:
	range3-end: ----->	{0.0.0.0}:
	range4-start: ----->	{0.0.0.0}:
	range4-end: ----->	{0.0.0.0}:
	default-lease-time: ---->	{-1}:
	min-lease-time: ----->	{-1}:
	max-lease-time: ----->	{-1}:
	boot-server: ----->	{0.0.0.0}:
	bootfile: ----->	{}
	default-router: ----->	{10.107.8.1}:
	primary-name-server: --->	{0.0.0.0}:
	secondary-name-server: ->	{0.0.0.0}:
	domain-name: ----->	{}
	subnetgroup: ----->	{2}:
	stickyaddr: ----->	{enable}:
	external-server: ----->	{0.0.0.p}:
	external-server-alt: --->	{0.0.0.0}:
		

Host-based routing with the MXK as a local DHCP server

When configuring host-based routing with the MXK as a local DHCP server, first create an floating IP interface, then create and configure a **dhcp-server-subnet** profile on the MXK.

The **dhcp-server-subnet** profile is associated with the floating IP interface to provide the IP address pool for the hosts. [Figure12](#) displays a MXK as a local DHCP server.

Figure 12: MXK as a local DHCP server



Configuring host-based routing with the MXK as a local DHCP server

- 1 Create a floating IP interface designating the IP address and subnet that will provide the IP addresses to all devices in the subnet.

```
zSH> interface add floatpmt2 10.107.8.254 255.255.255.0
Created ip-interface-record pmt2/ip.
```

Verify the interface with the **list ip-interface-record interface/type** command.

For large configurations, simply entering **list ip-interface-type** may display more information than is useful.

```
zSH> list ip-interface-record pmt2/ip
ip-interface-record pmt2/ip
1 entry found.
```

View the **ip-interface-record** profile for the floating IP interface.

```

zSH> get ip-interface-record 10.107.8.254/ip
ip-interface-record 10.107.8.254/ip
vpi: -----> {0}
vci: -----> {0}
rdindex: -----> {1}
dhcp: -----> {none}
addr: -----> {10.107.8.254} floating IP address
netmask: -----> {255.255.255.0} subnet mask
bcastaddr: -----> {10.107.8.255} broadcast address for the subnet
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmaxsize: -----> {0}
ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: --> {0}
egressPacketRuleGroupIndex: --> {0}
ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {false}
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {0}
maxVideoStreams: -----> {0}
tosOption: -----> {disable}
tosCOS: -----> {0}
vlanCOS: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}

```

2 Create the **dhcp-server-subnet** and reference the floating IP interface.

```

zSH> new dhcp-server-subnet 1
dhcp-server-subnet 1
Please provide the following: [q]uit.
network: -----> {0.0.0.0} 10.107.8.0 floating IP address
netmask: -----> {0.0.0.0} 255.255.255.0 subnet mask
domain: -----> {0}:
range1-start: -----> {0.0.0.0} 10.107.8.1 range of available
range1-end: -----> {0.0.0.0} 10.107.8.253 addresses
range2-start: -----> {0.0.0.0}:
range2-end: -----> {0.0.0.0}:
range3-start: -----> {0.0.0.0}:
range3-end: -----> {0.0.0.0}:
range4-start: -----> {0.0.0.0}:
range4-end: -----> {0.0.0.0}:
default-lease-time: ----> {-1}:

```

```

min-lease-time: -----> {-1}:
max-lease-time: -----> {-1}:
boot-server: -----> {0.0.0.0}:
bootfile: -----> {}:
default-router: -----> {0.0.0.0} 10.107.8.254 references the floating IP interface
primary-name-server: ---> {0.0.0.0}:
secondary-name-server: -> {0.0.0.0}:
domain-name: -----> {}:
subnetgroup: -----> {0} 1 subnet group number
stickyaddr: -----> {enable}:
external-server: -----> {0.0.0.0}:
external-server-alt: ---> {0.0.0.0}:
.....
Save new record? [s]ave, [c]hange or [q]uit:
New record saved.

```

- 3 Create the host interface. The *1* refers to the subnet group number *1*, and *5* designates the number of floating IP addresses allowed.

```

zSH> host add 1-13-1-0/eth dynamic 1 5
Adding host for 1-13-1-0/eth
zSH> host add 1-13-2-0/eth dynamic 1 5
Adding host for 1-13-2-0/eth

```

Verify the host interface by entering **host show interface**.

For large configurations, simply entering **host show** may display unneeded amounts of data.

```

zSH> host show 1-13-1-0-eth

```

Rd/Address	Interface	Group	T	Host	Address
1 10.107.8.254	1-13-1-0-eth	1	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	

```

zSH> host show 1-13-2-0-eth

```

Rd/Address	Interface	Group	T	Host	Address
1 10.107.8.254	1-13-2-0-eth	1	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	

Deleting the configuration

- 1 Delete the host(s). There are several ways to delete IP interfaces associated with an *interface/type*.

host delete <ip address> deletes the static host IP interface. See [Delete the static host IP interface. on page 135](#).

host delete unused <number> deletes the designated number of unassigned floating IP slots.

```
zSH> host delete 1-13-2-0/eth unused 4
Deleting host for 1-13-2-0/eth
```

host delete all deletes *all* of the host addresses on the designated interface, both assigned and unassigned.

```
zSH> host delete 1-13-1-0/eth all
Deleting host for 1-13-1-0/eth
```

2 Delete the **dhcp-server-subnet**.

The subnet will not be deleted if any provisioned interfaces are dependent on the subnet.

```
zSH> delete dhcp-server-subnet 1
dhcp-server-subnet 1
1 entry found.
Delete dhcp-server-subnet 1? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 1 deleted.
```

3 Delete the floating interface.

```
zSH> interface delete floatpmt2
Interface pmt2 deleted
```

Host-based routing with the MXK as a local DHCP server to provide DNS and bootp services

You can configure host-based routing with the MXK as a local DHCP server to provide DNS and bootp services.

Configuring host-based routing with an external DHCP server to provide DNS and bootp services

- 1 Create a floating IP interface designating the IP address and subnet that will provide the IP address to all devices in the subnet.

```
zSH> interface add float flt3 10.107.8.1/24 255.255.255.0
Created ip-interface-record flt3/ip.
```

Verify the interface with the **list ip-interface-record** *interface/type* command.

For large configurations, simply entering **list ip-interface-type** may display more information than is useful.

```
zSH> list ip-interface-record flt3/ip
ip-interface-record flt3/ip
1 entry found.
```

Verify the **ip-interface-record** profile for *flt3*.

```

zSH> get ip-interface-record flt3/ip
ip-interface-record flt3/ip
vpi: -----> {0}
vci: -----> {0}
rdindex: -----> {1}
dhcp: -----> {none}
addr: -----> {10.107.8.1} floating IP address
netmask: -----> {255.255.255.0} subnet mask
bcastaddr: -----> {10.107.8.255} broadcast address for subnet
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmmaxsize: -----> {0}
ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: -> {0}
egressPacketRuleGroupIndex: --> {0}
ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {false}
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {0}
maxVideoStreams: -----> {0}
tosOption: -----> {disable}
tosCOS: -----> {0}
vlanCOS: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}

```

- 2 Create the **dhcp-server-subnet** and specify the group number for the subnet, and enter the floating IP address, subnet mask, range of IP addresses to assign the hosts, the IP address of the boot server, the boot filename, and the primary and secondary IP addresses and domain name to be used by the DNS server.

```

zSH> new dhcp-server-subnet 3
dhcp-server-subnet 3
Please provide the following: [q]uit.
network: -----> {0.0.0.0}: 10.107.8.0
netmask: -----> {0.0.0.0}: 255.255.255.0
domain: -----> {0}:
range1-start: -----> {0.0.0.0}: 10.107.8.2
range1-end: -----> {0.0.0.0}: 10.107.8.250
range2-start: -----> {0.0.0.0}:
range2-end: -----> {0.0.0.0}:
range3-start: -----> {0.0.0.0}:

```

```
range3-end: -----> {0.0.0.0}:
range4-start: -----> {0.0.0.0}:
range4-end: -----> {0.0.0.0}:
default-lease-time: ----> {-1}:
min-lease-time: -----> {-1}:
max-lease-time: -----> {-1}:
boot-server: -----> {0.0.0.0}: 192.168.1.55
bootfile: -----> {}: filename.bin
default-router: -----> {0.0.0.0}: 10.107.8.1
primary-name-server: ---> {0.0.0.0}: 63.45.66.1
secondary-name-server: -> {0.0.0.0}: 63.45.66.1
domain-name: -----> {}: yourcompanyname.com
subnetgroup: -----> {0}: 3
stickyaddr: -----> {enable}:
external-server: -----> {0.0.0.0}:
external-server-alt: ---> {0.0.0.0}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Figure 13: DHCP server services available in the dhcp-server-subnet profile

floating IP address	network: ----->	{0.0.0.0}:10.107.8.0
subnet mask	netmask: ----->	{0.0.0.0}:255.255.255.0
	domain: ----->	{0}:
subnet ranges	range1-start: ----->	{0.0.0.0}:10.107.8.2
	range1-end: ----->	{0.0.0.0}:10.107.8.250
	range2-start: ----->	{0.0.0.0}:
	range2-end: ----->	{0.0.0.0}:
	range3-start: ----->	{0.0.0.0}:
	range3-end: ----->	{0.0.0.0}:
	range4-start: ----->	{0.0.0.0}:
	range4-end: ----->	{0.0.0.0}:
	default-lease-time: ---->	{-1}:
	min-lease-time: ----->	{-1}:
bootp services	max-lease-time: ----->	{-1}:
	boot-server: ----->	{0.0.0.0}:192.168.1.55
	bootfile: ----->	{ }filename.bin
DNS services	default-router: ----->	{10.107.8.1}:
	primary-name-server: --->	{0.0.0.0}:63.45.66.1
	secondary-name-server: ->	{0.0.0.0}:63.45.66.1
	domain-name: ----->	{ }yourcompanyname.com
	subnetgroup: ----->	{2}:
	stickyaddr: ----->	{enable}:
	external-server: ----->	{0.0.0.p}:
	external-server-alt: --->	{0.0.0.0}:
		

3 Verify the entries for **dhcp-server-subnet 3**.

```
zSH> get dhcp-server-subnet 3
dhcp-server-subnet 3
network: -----> {10.107.8.0}
```

```

netmask: -----> {255.255.255.0}
domain: -----> {0}
range1-start: -----> {10.107.8.2}
range1-end: -----> {10.107.8.250}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: ----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {192.168.1.55}
bootfile: -----> {filename.bin}
default-router: -----> {10.107.8.1}
primary-name-server: ---> {63.45.66.1}
secondary-name-server: -> {63.45.66.1}
domain-name: -----> {yourcompanyname.com}
subnetgroup: -----> {3}
stickyaddr: -----> {enable}
external-server: -----> {0.0.0.0}
external-server-alt: ---> {0.0.0.0}

```

- 4 Create the host route and designate which subnet group you want to associate with the host. The 3 refers to the subnet group 3 defined when creating the **dhcp-server-subnet**, and 2 designates the number of floating IP addresses allowed.

```

zSH> host add 1-13-4-0/eth dynamic 3 2
Adding host for 1-13-4-0/eth

```

Verify the host interface by entering **host show interface**. For large configurations, simply entering **host show** may display unneeded amounts of data.

```

zSH> host show 1-13-4-0-eth

```

Rd/Address	Interface	Group	T	Host	Address
1 10.107.8.1	1-13-4-0-eth	3	D	<unassigned>	
			D	<unassigned>	

Deleting the configuration

- 1 Delete the host.

```

zSH> host delete 1-13-4-0/eth unused 2
Deleting host for 1-13-4-0/eth

```

- 2 Delete the **dhcp-server-subnet**.

```

zSH> delete dhcp-server-subnet 3
dhcp-server-subnet 3
1 entry found.

```

```
Delete dhcp-server-subnet 3? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 3 deleted.
```

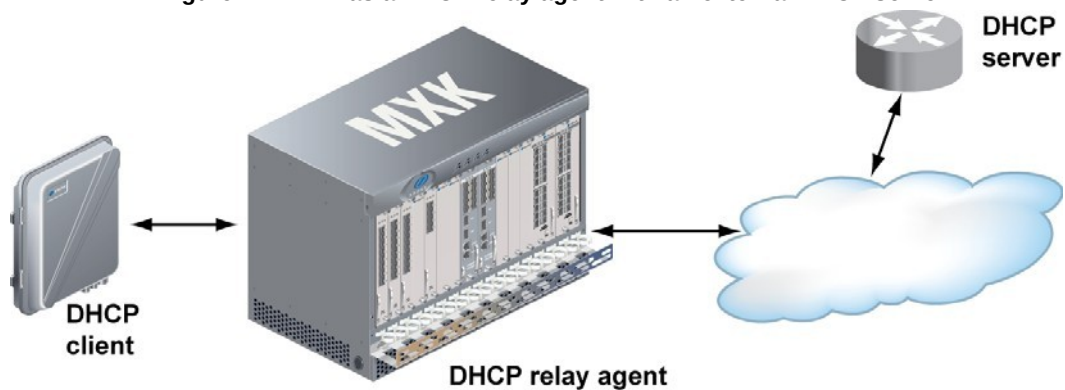
3 Delete the floating interface.

```
zSH> interface delete float flt3
Interface flt3 deleted
```

Host-based routing with an external DHCP server

Host-based routing on the MXK with an external DHCP server, configures the MXK to relay traffic between the hosts and the DHCP server. [Figure 14](#) shows the MXK as a DHCP relay agent with an external DHCP server.

Figure 14: MXK as a DHCP relay agent with an external DHCP server



Note: You can configure the MXK either as a local DHCP server or configure the MXK to use an external DHCP server. The MXK cannot be a local DHCP server and use an external DHCP on the same subnet.

However, you can use the MXK as a local DHCP server and have an external DHCP if the subnets are not the same.

Configuring the MXK for host-based routing with an external DHCP server

- 1 Create a floating IP interface designating the IP address and subnet that will provide the IP addresses to all devices in the subnet.

```
zSH> interface add float flt1 192.168.49.1 255.255.255.0
Created ip-interface-record flt1/ip.
```

Verify the interface with the **list ip-interface-record interface/type** command.

For large configurations, simply entering **list ip-interface-type** may display more information than is useful.

```
zSH> list ip-interface-record flt1/ip
```



```
ip-interface-record f1t1/ip
1 entry found.
```

View the **ip-interface-record** profile for *f1t1/ip*.

```
zSH> get ip-interface-record f1t1/ip
ip-interface-record f1t1/ip
vpi: -----> {0}
vci: -----> {0}
rdindex: -----> {1}
dhcp: -----> {none}
addr: -----> {192.168.49.1} floating ip address
netmask: -----> {255.255.255.0} subnet mask
bcastaddr: -----> {192.168.49.255} broadcast address for the subnet
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmaxsize: -----> {0}
ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: -> {0}
egressPacketRuleGroupIndex: --> {0}
ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {false}
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {0}
maxVideoStreams: -----> {0}
tosOption: -----> {disable}
tosCOS: -----> {0}
vlanCOS: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}
```

- 2 Create the DHCP relay agent by entering the IP address of the DHCP server and associating the floating IP interface with the DHCP server with the **dhcp-relay add** *<ip-address>* *<interface>* command.

```
zSH> dhcp-relay add 192.168.88.73 f1t1
Created DHCP Relay Agent number 1
```

This command creates the **dhcp-server-subnet** profile that defines the DHCP relay agent and assigns the subnet group the first available group number, in this case *1*.

Verify the dhcp-relay agent and the subnet group number.

```
zSH> list dhcp-server-subnet 1
dhcp-server-subnet 1
1 entry found.
```

View the **dhcp-server-subnet**.

```
zSH> get dhcp-server-subnet 1
dhcp-server-subnet 1
network: -----> {192.168.49.0}  network address
netmask: -----> {255.255.255.0}  subnet mask
domain: -----> {0}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: -----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {192.168.49.1}  references the floating IP address
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {1}  the system assigned subnet group number
stickyaddr: -----> {enable}
external-server: -----> {194.168.88.73}  references the external DHCP server
external-server-alt: ---> {0.0.0.0}
```

- 3 Create the host route. The 1 refers to the subnet group 1 you defined when creating the **dhcp-server-subnet**, and 3 designates the number of floating IP addresses allowed for the host.

```
zSH> host add 1-13-5-0/eth dynamic 1 3
Adding host for 1-13-5-0/eth
```

Verify the host interface by entering **host show interface**. For large configurations, simply entering **host show** may display unneeded amounts of data.

```
zSH> host show 1-13-5-0-eth
Rd/Address      Interface      Group T Host Address
-----
1 192.168.49.1  1-13-5-0-eth  1      D <unassigned>
                  D <unassigned>
                  D <unassigned>
```

Deleting the configuration

- 1 When necessary, delete the host.

```
zSH> host delete 1-13-1-0/eth unused 3
Deleting host for 1-13-1-0/eth
```

2 Delete the dhcp-server subnet.

```
zSH> delete dhcp-server-subnet 1
dhcp-server-subnet 1
1 entry found.
Delete dhcp-server-subnet 1? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 1 deleted.
```

3 Delete the floating IP interface.

```
zSH> interface delete float flt1
Interface flt1 deleted
```

Host-based routing with multiple dhcp-relay agents and one DHCP server

Configuring host-based routing with an external DHCP server and multiple dhcp-relay agents creates additional floating IP addresses.

Some configurations need more than one floating IP address or need large numbers of subnets.

Configuring host-based routing with an external DHCP server and multiple dhcp-relay agents

- 1 Create more than one floating IP interfaces by designating the IP addresses and subnets that will provide the IP addresses to all of the devices in each subnet.

```
zSH> interface add float flt1 10.101.8.1/24 255.255.255.0
Created ip-interface-record flt1/ip.
```

Verify the interface with the **list ip-interface-record interface/type** command.

For large configurations, simply entering **list ip-interface-type** may display more information than is useful.

```
zSH> list ip-interface-record flt1/ip
ip-interface-record flt1/ip
1 entry found.
```

Create another floating IP interface.

```
zSH> interface add float flt2 10.102.8.1/24 255.255.255.0
Created ip-interface-record flt2/ip.
```

Verify the IP interface.

```
zSH> list ip-interface-record flt2/ip
ip-interface-record flt2/ip
```

1 entry found.

- 2 Create the dhcp-server relay agent by entering the IP address of the DHCP server and associating the floating IP interface with the DHCP server with the **dhcp-relay add** *<ip address>* *<interface>* command.

```
zSH> dhcp-relay add 192.168.88.73 flt1
Created DHCP Relay Agent number 1
```

This command creates the **dhcp-server-subnet** profile that defines the DHCP relay agent and assigns the subnet group the first available group number, in this case *1*.

Verify the **dhcp-server-subnet**.

```
zSH> list dhcp-server-subnet 1
dhcp-server-subnet 1
1 entry found.
```

View the **dhcp-server-subnet 1** profile.

```
zSH> get dhcp-server-subnet 1
dhcp-server-subnet 1
network: -----> {10.101.8.0} network address
netmask: -----> {255.255.255.0} subnet mask
domain: -----> {0}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: ----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {10.101.8.1} references the floating IP address
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {1} system assigned subnet group number
stickyaddr: -----> {enable}
external-server: -----> {192.168.88.73} references the external DHCP server
external-server-alt: ---> {0.0.0.0}
```

- 3 Create the next DHCP relay agent by entering the same IP address for the DHCP server and associate a different floating IP interface with the DHCP server using the **dhcp-relay add** *<ip-address>* *<interface>* command.

```
zSH> dhcp-relay add 192.168.88.73 flt2
Created DHCP Relay Agent number 2
```

This command creates the **dhcp-server-subnet** profile that defines the DHCP relay agent and assigns the subnet group the first available group number, in this case 2.

Verify the **dhcp-server-subnet**.

```
zSH> list dhcp-server-subnet 2
dhcp-server-subnet 2
1 entry found.
```

View the **dhcp-server-subnet 2** profile.

```
zSH> get dhcp-server-subnet 2
dhcp-server-subnet 2
network: -----> {10.102.8.0} network address
netmask: -----> {255.255.255.0} subnet mask
domain: -----> {0}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: ----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {10.102.8.1} references the floating IP address
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {2} the system assigned subnet group number
stickyaddr: -----> {enable}
external-server: -----> {192.168.88.73} references the external DHCP server
external-server-alt: ---> {0.0.0.0}
```

- 4 Create the host route and designate which subnet group to associate with the host. The 1 refers to the subnet group 1 defined when creating the **dhcp-server-subnet**, and 2 designates the number of floating IP addresses allowed.

```
zSH> host add 1-13-1-0/eth dynamic 1 2
Adding host for 1-13-1-0/eth
```

Create the next host route designating the subnet group 2 and the number of floating IP addresses allowed.

```
zSH> host add 1-13-2-0/eth dynamic 2 2
Adding host for 1-13-2-0/eth
```

Verify the host interface by entering **host show interface**. For large configurations, simply entering **host show** may display unneeded amounts of data.

zSH> host show1-13-1-0-eth

Rd/Address	Interface	Group	T	Host	Address
1 10.101.8.1	1-13-1-0-eth	1	D	<unassigned>	D <unassigned>

zSH> host show1-13-2-0-eth

Rd/Address	Interface	Group	T	Host	Address
1 10.102.8.1	1-13-2-0-eth	2	D	<unassigned>	D <unassigned>

Deleting the configuration

- 1 Delete the host.

```
zSH> host delete 1-13-2-0/eth unused 2
Deleting host for 1-13-2-0/eth
```

- 2 Delete the **dhcp-server subnet**.
- 3 Delete the floating interface.

```
zSH> interface delete float flt2
Interface flt2 deleted
```

Host-based routing with an external DHCP server and an alternate DHPC server with dhcp-relay agent

You can use the **dhcp-relay add** command using the *alt* variable to designate a DHCP server and an alternate DHCP server for the same subnet.

Configuring host-based routing with an external DHCP server and an alternate DHCP server with dhcp-relay agent

- 1 Create a floating IP interface designating the IP address and subnet that will provide the IP addresses to all devices in the subnet.

```
zSH> interface add float flt4 10.103.8.1/24 255.255.255.0
Created ip-interface-record flt4/ip.
```

Verify the interface with the **list ip-interface-record interface/type** command.

For large configurations, simply entering **list ip-interface-type** may display more information than is useful.

```
zSH> list ip-interface-record flt4/ip
ip-interface-record flt4/ip
1 entry found.
```

- 2 Create the dhcp-server relay agent by designating the IP address of the DHCP server, the IP address of the alternate DHCP server, along with the floating IP interface.

```
zSH> dhcp-relay add 192.168.88.73 alt 192.168.87.74 flt4
Created DHCP Relay Agent number 3
```

The DHCP relay agent is created with a DHCP server subnet group number of 3.

- 3 Verify the **dhcp-server-subnet**.

```
zSH> get dhcp-server-subnet 3
dhcp-server-subnet 3
network: -----> {10.103.8.0} network address
netmask: -----> {255.255.255.0} subnet mask
domain: -----> {0}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: ----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {10.103.8.1} references the floating IP address
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {3} system assigned subnet group number
stickyaddr: -----> {enable}
external-server: -----> {192.168.88.73} references the external DHCP server
external-server-alt: ---> {192.168.87.74} references the alternate external DHCP server
```

- 4 Create the host route and designate which subnet group you want to associate with the host. The 2 refers to the subnet group 2 you defined when creating the **dhcp-server-subnet**, and 3 designates the number of floating IP addresses allowed.

```
zSH> host add 1-13-1-0/eth dynamic 3 2
Adding host for 1-13-1-0/eth
```

Verify the host interface by entering **host show interface**.

For large configurations, simply entering **host show** may display unneeded amounts of data.

```
zSH> host show 1-13-1-0-eth
Rd/Address      Interface      Group T Host Address
-----
```

1	10.103.8.1	1-13-1-0-eth	3	D <unassigned>
				D <unassigned>

Deleting the configuration

- 1 When necessary, delete the host.

```
zSH> host delete 1-13-1-0/eth all
Deleting host for 1-13-1-0/eth
```

- 2 Delete the dhcp-server subnet.

```
zSH> delete dhcp-server-subnet 2
dhcp-server-subnet 2
1 entry found.
Delete dhcp-server-subnet 2? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 2 deleted.
```

- 3 Delete the floating interfaces.

```
zSH> interface delete float flt1
Interface flt1 deleted
```

Network-based routing

This section discusses:

- [Network-based routing without DHCP, page153](#)
- [Network-based routing with the MXK as local DHCP server, page154](#)
- [Network-based routing with an external DHCP server, page156](#)

Network-based routing on an interface assigns one IP address to the interface along with the subnet routing table. Network-based routing on an interface assigns one IP address and the entire subnet to the interface and allows a single routing table entry. The subnet masks can be of variable lengths.

You can configure network-based routing on the MXK in one of three ways:

- configuration without a DHCP server.
See [Network-based routing without DHCP on page153](#)
- DHCP services are on the MXK (the MXK is the DHCP server).
[Host-based routing with the MXK as a local DHCP server on page137](#)
- The MXK uses an external DHCP server.
[Host-based routing with an external DHCP server on page144](#)

Refer to the *CLI Reference Guide* for a complete description of the command options and syntax.

Network-based routing without DHCP

Configuring network-based routing without DHCP

Create a point-to-point connection on an Ethernet interface that provides two IP addresses, one for the Ethernet interface and one for the downstream device. Ending the IP address in /24 specifies the two addresses.

- 1 Create an IP interface on an Ethernet uplink port for the upstream connection.

```
zSH> interface add 1-a-2-0/eth 192.169.1.14/24
Created ip-interface-record ethernet2/ip.
```

Add a route with a cost of two.

```
zSH> route add default 192.169.1.254 2
```

Verify the interface.

```
zSH> interface show
2 interfaces
```

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/a/1/0/ip	UP	1 172.16.160.49/24	00:01:47:17:ee:54	ethernet1
1/a/2/0/ip	DOWN	1 192.169.1.14/24	00:01:47:17:ee:55	ethernet2-777

- 2 Create an IP interface on an Ethernet port.

```
zSH> interface add 1-13-8-0/eth 172.24.1.1/24
Created ip-interface-record 1-13-8-0-eth/ip.
```

- 3 Verify the interface.

```
zSH> interface show
4 interfaces
```

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/13/6/0/ip	DOWN	1 172.25.1.1/30	00:01:47:1a:fe:91	1-13-6-0-eth
1/13/8/0/ip	DOWN	1 172.24.1.1/30	00:01:47:1a:fe:93	1-13-8-0-eth
1/a/1/0/ip	UP	1 172.16.160.49/24	00:01:47:17:ee:54	ethernet1
1/a/2/0/ip	DOWN	1 192.169.1.14/24	00:01:47:17:ee:55	ethernet2-777

- 4 View the **ip-interface-record** profile.

In this case, note that the **dhcp** parameter is set to *none* and the **dhcpserverenable** parameter is set to *false* in the **ip-interface-record** profile. This interface cannot provide DHCP services.

```
zSH> get ip-interface-record 1-13-8-0-eth/ip
ip-interface-record 1-13-8-0-eth/ip
vpi: -----> {0}
vci: -----> {0}
```

```

rdindex: -----> {1}
dhcp: -----> {none} DHCP services not provided
addr: -----> {172.24.1.1}
netmask: -----> {255.255.255.0}
bcastaddr: -----> {172.24.1.255}
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmaxsize: -----> {0}
ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: -> {0}
egressPacketRuleGroupIndex: --> {0}
ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {false} DHCP services not provided
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {0}
maxVideoStreams: -----> {0}
tosOption: -----> {disable}
tosCOS: -----> {0}
vlanCOS: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}

```

Deleting the network-based routing configuration

Delete the IP interface on the Ethernet port.

```

zSH> interface delete 1-13-8-0-eth/ip
Delete complete

```

The **ip-interface-record** profile is deleted from the system.

Network-based routing with the MXK as local DHCP server

You can configure the MXK to act as a local DHCP server in a network-based routing configuration.

Configuring network-based routing with MXK as local DHCP server

Specifying *server* in the CLI enables the DHCP server functionality locally on the MXK. However, services such as DNS or bootp are not enabled.

- 1 Create an IP interface on an Ethernet uplink port.

```
zSH> interface add 1-a-2-0/eth vlan 777 192.169.1.14/24
Created ip-interface-record ethernet2-777/ip.
```

Add a route with a cost of one.

```
zSH> route add default 192.169.1.254 1
```

Verify the interface.

```
zSH> interface show
2 interfaces
```

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/a/1/0/ip	UP	1 172.16.160.49/24	00:01:47:17:ee:54	ethernet1
1/a/2/0/ip	DOWN	1 192.169.1.14/24	00:01:47:17:ee:55	ethernet2-777

- 2 Create the IP interface on an Ethernet port.

```
zSH> interface add 1-13-9-0/eth 172.26.1.1/10 server
Created ip-interface-record 1-13-9-0-eth/ip.
```

The **ip-interface-record** profile is created with the DHCP server functionality enabled. See [Step 4](#).

- 3 Verify the interface.

```
zSH> interface show
5 interfaces
```

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/13/8/0/ip	DOWN	1 172.24.1.1/30	00:01:47:1a:fe:93	1-13-8-0-eth
1/13/9/0/ip	DOWN	1 172.26.1.1/10	00:01:47:1a:fe:94	1-13-9-0-eth
1/a/1/0/ip	UP	1 172.16.160.49/24	00:01:47:17:ee:54	ethernet1
1/a/2/0/ip	DOWN	1 192.169.1.14/24	00:01:47:17:ee:55	ethernet2-777

- 4 View the **ip-interface-record** profile to verify that the DHCP server functionality is enabled.

In this case, note that the **dhcp** parameter is set to *server* and the **dhcpserverenable** parameter is set to *true* in the **ip-interface-record** profile. This interface now provides basic DHCP services.

```
zSH> get ip-interface-record 1-13-9-0-eth/ip
ip-interface-record 1-13-9-0-eth/ip
vpi: -----> {0}
vci: -----> {0}
```

```

rdindex: -----> {1}
dhcp: -----> {server} DHCP server function is enabled
addr: -----> {172.26.1.1}
netmask: -----> {255.255.255.252}
bcastaddr: -----> {172.26.1.3}
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmaxsize: -----> {0}
ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: --> {0}
egressPacketRuleGroupIndex: --> {0}
ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {true} DHCP server function is enabled
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {0}
maxVideoStreams: -----> {0}
tosOption: -----> {disable}
tosCOS: -----> {0}
vlanCOS: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}

```

Deleting the configuration

When necessary, delete the IP interface on the Ethernet port.

```

zSH> interface delete 1-13-9-0-eth/ip
Delete complete

```

The **ip-interface-record** profile is deleted from the system.

Network-based routing with an external DHCP server

The MXK acts as a DHCP relay agent to an external DHCP server.

Configuring network-based routing with external DHCP server

- 1 Create an IP interface on an Ethernet uplink port:

```

zSH> interface add 1-a-2-0/eth vlan 777 192.169.1.14/24

```

Created ip-interface-record ethernet2-777/ip.

Add a route with a cost of one.

```
zSH> route add default 192.169.1.254 1
```

Verify the interface.

```
zSH> interface show
```

2 interfaces

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/a/1/0/ip	UP	1 172.16.160.49/24	00:01:47:17:ee:54	ethernet1
1/a/2/0/ip	DOWN	1 192.169.1.14/24	00:01:47:17:ee:55	ethernet2-777

2 Create an IP interface on an Ethernet port.

```
zSH> interface add 1-13-19-0/eth 10.109.8.1/24
```

Created ip-interface-record 1-13-19-0-eth/ip.

3 Create the dhcp-server relay agent by designating the IP address of the DHCP server and associating the relay agent with the IP interface.

```
zSH> dhcp-relay add 172.24.72.102 1-13-19-0-eth/ip
```

Created DHCP Relay Agent number 3

4 View the dhcp-server-subnet profile.

```
zSH> get dhcp-server-subnet 3
```

dhcp-server-subnet 3

```
network: -----> {10.109.8.0}
netmask: -----> {255.255.255.0}
domain: -----> {0}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: ----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {10.109.8.1}
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {3}
stickyaddr: -----> {enable}
external-server: -----> {172.24.72.102}
external-server-alt: ---> {0.0.0.0}
```

Deleting the network-based routing configuration

- 1 When necessary, delete the dhcp-server relay agent.

```
zSH> dhcp-relay delete 3
Deleted DHCP Relay Agent number 3
```

The **dhcp-server-subnet 3** profile is deleted.

- 2 Delete the IP interface.

```
zSH> interface delete 1-13-19-0-eth/ip
Delete complete
```

Host-based routing for triple-play services on Ethernet

This section describes the steps to create host-based routing for triple-play services on Ethernet. For more information on routed video services, see [Chapter 6, Video Configuration, on page 315](#).

To configure the MXK for triple play services (voice, video, and data), create three different floating IP interfaces, one for each service.

Creating host-based routing for triple-play services on Ethernet

- 1 Create an IP interface on an Ethernet uplink port.

```
zSH> interface add 1-a-2-0/eth 192.169.1.14/24
Created ip-interface-record ethernet2-777/ip.
```

Add a route with a cost of one.

```
zSH> route add default 192.169.1.254 1
```

Verify the interface.

```
zSH> interface show
3 interfaces
```

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/a/2/0/ip	DOWN	1 192.169.1.14/24	00:01:47:1a:fe:65	ethernet2
1/a/4/0/ip	UP	1 172.16.7.49/24	00:01:47:1a:fe:67	ethernet4-7
1/a/6/0/ip	UP	1 172.16.160.49/24	00:01:47:1a:fe:64	ipobridge-160

- 2 Create a **video-source** profile for video services. Enter the IP interface connected to the video source.

```
zSH> new video-source 1
video-source 1
Please provide the following: [q]uit.
routing-domain: ----> {0}: 1
multicast-address: -> {0.0.0.0}: 224.1.1.1
ifIndex: -----> {0/0/0/0/0}: ethernet2/ip
```

```
vpi: -----> {0}:
vci: -----> {0}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```



Note: You only need to enter the first multicast address in the group.

View the **video-source** profile.

```
zSH> get video-source 1
video-source 1
routing-domain: ----> {1}
multicast-address: -> {224.1.1.1}
ifIndex: -----> {ethernet2/ip}
vpi: -----> {0}
vci: -----> {0}
```

- 3 Create a multicast control list for video services. The first digit defines the video package and the second digit defines the channel. The IP address associates a video stream for the channel.

```
zSH> new mcast-control-entry 1/1
mcast-control-entry 1/1
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.1
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Verify the multicast control list.

```
zSH> list mcast-control-entry
mcast-control-entry 1/1
1 entry found.
```

- 4 Create a floating IP interface for each service. The IP interface designates the IP address and subnet that will provide the IP addresses to all the devices in the subnet.

- a Create the floating IP interface for data services.

```
zSH> interface add float flt1 192.168.49.1/24 255.255.255.0
Created ip-interface-record flt1/ip.
```

- b Create the floating IP interface for voice services.

```
zSH> interface add float flt2 192.168.48.1/24 255.255.255.0
Created ip-interface-record flt2/ip.
```

- c Create the IP interface for video services.

```
zHS> interface add float f1t3 192.168.47.1/24 255.255.255.0
Created ip-interface-record f1t3/ip.
```

d Verify the floating IP interfaces.

```
zSH> list ip-interface-record f1t1/ip
ip-interface-record f1t1/ip
1 entry found.
```

```
zSH> list ip-interface-record f1t2/ip
ip-interface-record f1t2/ip
1 entry found.
```

```
zSH> list ip-interface-record f1t3/ip
ip-interface-record f1t3/ip
1 entry found.
```

5 Create the dhcp-server relay agent for each service by designating the IP address of the DHCP server that will provide the services and the floating IP interface.

a Provide the IP address of the external DHCP server that is providing data services.

```
zSH> dhcp-relay add 192.168.88.73 f1t1
Created DHCP Relay Agent number 1
```

b Create a dhcp-server relay agent for voice services.

```
zSH> dhcp-relay add 192.168.89.73 f1t2
Created DHCP Relay Agent number 2
```

c Create a dhcp-relay agent for video services.

```
zSH> dhcp-relay add 192.168.87.73 f1t3
Created DHCP Relay Agent number 3
```

d Verify the **dhcp-server-subnet(s)**.

```
zSH> list dhcp-server-subnet
dhcp-server-subnet 1
dhcp-server-subnet 2
dhcp-server-subnet 3
3 entries found.
```

6 Create the host routes for the triple-play services. Assign a separate VLAN ID for each service. These VLANs are terminated at the interface. VLANs should match VLANs configured on the CPE devices.

a Add a host route for data services.

The *1* refers to the **dhcp-server-subnet** group and the *5* refers to the number of floating IP addresses allowed.

```
zSH> host add 1-13-1-0/eth vlan 100 dynamic 1 5
Adding host for 1-13-1-0/eth
```


b Add a host route for voice services.

The 2 refers to the **dhcp-server-subnet** group and the 1 refers to the number of floating IP addresses allowed.

```
zSH> host add 1-13-2-0/eth vlan 200 dynamic 2 1
Adding host for 1-13-2-0/eth
```

c Add a host route for video services.

The 3 refers to the **dhcp-server-subnet** group and the 3 refers to the number of floating IP addresses assigned for set-top boxes. For video services, *video 1/5* sets the multicast control list index and the maximum number of IP video streams.

```
zSH> host add 1-13-3-0/eth vlan 300 dynamic 3 3 video 1/5
Adding host for 1-13-3-0/eth
```

d Verify the hosts.

```
zSH> host show 1-13-1-0-eth-100
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.49.1	1-13-1-0-eth-100	1	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	

```
zSH> host show 1-13-2-0-eth-200
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.48.1	1-13-2-0-eth-200	2	D	<unassigned>	

```
zSH> host show 1-13-3-0-eth-300
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.47.1	1-13-3-0-eth-300	3	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	

For more information on configuring video, see [Chapter 6, Video Configuration](#), on page 315.

Deleting the triple-play configuration

Delete the triple-play configuration.

1 Delete the host routes.

```
zSH> host delete 1-13-1-0/eth vlan 100 all
Deleting host for 1-13-1-0/eth
```

```
zSH> host delete 1-13-2-0/eth vlan 200 all
Deleting host for 1-13-2-0/eth
```

```
zSH> host delete 1-13-3-0/eth vlan 300 all
Deleting host for 1-13-3-0/eth
```

- 2 Delete the **dhcp-server-subnet** profiles created with the **dhcp-relay add** command.

```
zSH> delete dhcp-server-subnet 1
dhcp-server-subnet 1
1 entry found.
Delete dhcp-server-subnet 1? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 1 deleted.
```

```
zSH> delete dhcp-server-subnet 2
dhcp-server-subnet 2
1 entry found.
Delete dhcp-server-subnet 2? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 2 deleted.
```

```
zSH> delete dhcp-server-subnet 3
dhcp-server-subnet 3
1 entry found.
Delete dhcp-server-subnet 3? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 3 deleted.
```

- 3 Delete the floating IP interfaces.

```
zSH> delete ip-interface-record f1t1/ip
ip-interface-record f1t1/ip
1 entry found.
Delete ip-interface-record f1t1/ip? [y]es, [n]o, [q]uit : y
ip-interface-record f1t1/ip deleted.
```

```
zSH> delete ip-interface-record f1t2/ip
ip-interface-record f1t2/ip
1 entry found.
Delete ip-interface-record f1t2/ip? [y]es, [n]o, [q]uit : y
ip-interface-record f1t2/ip deleted.
```

```
zSH> delete ip-interface-record f1t3/ip
ip-interface-record f1t3/ip
1 entry found.
Delete ip-interface-record f1t3/ip? [y]es, [n]o, [q]uit : y
ip-interface-record f1t3/ip deleted.
```

- 4 Delete the multicast control list.

```
zSH> delete mcast-control-entry 1/1
mcast-control-entry 1/1
1 entry found.
Delete mcast-control-entry 1/1? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/1 deleted.
```

- 5 Delete the video source.

```
zSH> delete video-source 1
```

```

video-source 1
1 entry found.
Delete video-source 1? [y]es, [n]o, [q]uit : y
video-source 1 deleted.

```

Host-based routing for triple-play services on GPON

This section explains how to configure the MXK for triple play services (voice, video, and data) on GPON. For triple-play services you would want to create three different floating IP interfaces for the different services. Typically, you need public IP addresses for data services, and private IP addresses for video and VoIP services.



Note: For information on Smart OMCI and ONU management, see [Chapter 9, MXK GPON Cards](#). For more information on configuring routed video on the MXK, see [Chapter 6, Video Configuration](#).

Creating host-based routing for triple-play services on GPON

- 1 Create an IP interface on an Ethernet uplink port.

```

zSH> interface add 1-a-2-0/eth vlan 777 192.169.1.14/24
Created ip-interface-record ethernet2-777/ip.

```

Add a route with a cost of one.

```

zSH> route add default 192.169.1.254 1

```

Verify the interface.

```

zSH> interface show
2 interfaces

```

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/a/1/0/ip	UP	1 172.16.160.49/24	00:01:47:17:ee:54	ethernet1
1/a/2/0/ip	DOWN	1 192.169.1.14/24	00:01:47:17:ee:55	ethernet2-777

- 2 Create the **video-source** profile for video services.

```

zSH> new video-source 1
video-source 1
Please provide the following: [q]uit.
routing-domain: ----> {0}: 1
multicast-address: -> {0.0.0.0}: 224.1.1.1
ifIndex: -----> {0/0/0/0/0}: ethernet2-777/ip
vpi: -----> {0}:
vci: -----> {0}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

```



Note: You only need to enter the first multicast address in the group.

View the **video-source** profile.

```
zSH> get video-source 1
video-source 1
routing-domain: ----> {1}
multicast-address: -> {224.1.1.1}
ifIndex: -----> {ethernet2-777/ip}
vpi: -----> {0}
vci: -----> {0}
```

3 Create a multicast control list for video services.

```
zSH> new mcast-control-entry 1/1
mcast-control-entry 1/1
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.1
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Verify the multicast control list.

```
zSH> list mcast-control-entry
mcast-control-entry 1/1
1 entry found.
```

4 Create the GPON traffic descriptors for the GPON triple-play services.

```
zSH> new gpon-traffic-profile 1
gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}:
traffic-class: -----> {ubr}:
compensated: -----> {false}:
shared: -----> {false}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
zSH> new gpon-traffic-profile 2
gpon-traffic-profile 2
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}:
traffic-class: -----> {ubr}:
compensated: -----> {false}:
shared: -----> {false}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
zSH> new gpon-traffic-profile 3
gpon-traffic-profile 3
```

```

Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}:
traffic-class: -----> {ubr}:
compensated: -----> {false}:
shared: -----> {false}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

```

Verify the traffic profiles.

```

zSH> list gpon-traffic-profile
gpon-traffic-profile 1
gpon-traffic-profile 2
gpon-traffic-profile 3
3 entries found.

```

- 5 Create floating IP interfaces designating the IP address and subnet that will provide the IP addresses to all the devices in the subnet.

Designate public or private IP addresses for the floating IP interface, depending on the services provided.

- a Provide public IP addresses for data services.

```

zSH> interface add float flt1 192.168.49.1 255.255.255.0
Created ip-interface-record flt1/ip.

```

- b Provide private IP addresses for voice services.

```

zSH> interface add float flt2 10.107.8.1/24 255.255.255.0
Created ip-interface-record flt2/ip.

```

- c Provide private IP addresses for video services.

```

zSH> interface add float flt3 10.108.8.1/24 255.255.255.0
Created ip-interface-record flt3/ip.

```

- d Verify the floating IP interfaces.

```

zSH> list ip-interface-record
ip-interface-record ethernet1/ip
ip-interface-record ethernet2-777/ip
ip-interface-record flt1/ip
ip-interface-record flt2/ip
ip-interface-record flt3/ip
5 entries found.

```

- 6 Create the dhcp-server relay agent for each service by designating the IP address of the DHCP server and the floating IP interface.

- a Create a dhcp-server relay agent for data services.

```

zSH> dhcp-relay add 192.168.88.73 flt1
Created DHCP Relay Agent number 1

```

- b** Create a dhcp-server relay agent for voice services.

```
zSH> dhcp-relay add 192.168.89.73 flt2
Created DHCP Relay Agent number 2
```

- c** Create a dhcp-relay agent for video services.

```
zSH> dhcp-relay add 192.168.87.73 flt3
Created DHCP Relay Agent number 3
```

- d** Verify the dhcp-server-subnet(s).

```
zSH> list dhcp-server-subnet
dhcp-server-subnet 1
dhcp-server-subnet 2
dhcp-server-subnet 3
3 entries found.
```

- 7** Create the host routes for the triple-play services. Assign separate VLAN ID for each service.

This example configures GEM 501 for data services, GEM 701 for voice services, and GEM 901 for video services. The numbers 1, 2, and 3 refer to the DHCP subnet groups and 3 refers to the number of floating IP addresses allowed. For video services, *video 1/5* sets the multicast control list index and the maximum number of IP video streams.

- a** Add a host route for data services.

```
zSH> host add 1-9-1-501/gponport gtp 1 vlan 100 dynamic 1 3
GEM Port 1-9-1-501/gponport has been created on ONU 1-9-1-1/gpononu.
Adding host for 1-9-1-501/gponport
```

- b** Add a host route for voice services.

```
zSH> host add 1-9-1-701/gponport gtp 2 vlan 200 dynamic 2 3
GEM Port 1-9-1-701/gponport has been created on ONU 1-9-1-1/gpononu.
Adding host for 1-9-1-701/gponport
```

- c** Add a host route for video services.

```
zSH> host add 1-9-1-901/gponport gtp 3 vlan 300 dynamic 3 3 video 1/5
GEM Port 1-9-1-901/gponport has been created on ONU 1-9-1-1/gpononu.
Adding host for 1-9-1-901/gponport
```

- d** Verify the hosts.

```
zSH> host show
```

Rd/Address	Interface	Group	T	Host	Address
1 192.168.49.1	1-9-1-501-gponport-100	1	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
1 10.107.8.1	1-9-1-701-gponport-200	2	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	

1	10.108.8.1	1-9-1-901-gponport-300	3	D <unassigned>
				D <unassigned>
				D <unassigned>

For more information on configuring video, see [Chapter 6, Video Configuration, on page 315](#).

Deleting the triple-play configuration

When necessary, you can delete the triple-play configuration.

1 Delete the host routes.

```
zSH> host delete 1-9-1-501/gponport vlan 100 all
Deleting host for 1-9-1-501/gponport
GEM Port 1-9-1-501/gponport has been deleted.
```

```
zSH> host delete 1-9-1-701/gponport vlan 200 all
Deleting host for 1-9-1-701/gponport
GEM Port 1-9-1-701/gponport has been deleted.
```

```
zSH> host delete 1-9-1-901/gponport vlan 300 all
Deleting host for 1-9-1-901/gponport
GEM Port 1-9-1-901/gponport has been deleted.
```

2 Delete the **dhcp-server-subnet** profiles created with the **dhcp-relay add** command.

```
zSH> delete dhcp-server-subnet 1
dhcp-server-subnet 1
1 entry found.
Delete dhcp-server-subnet 1? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 1 deleted.
zSH> delete dhcp-server-subnet 2
dhcp-server-subnet 2
1 entry found.
Delete dhcp-server-subnet 2? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 2 deleted.
zSH> delete dhcp-server-subnet 3
dhcp-server-subnet 3
1 entry found.
Delete dhcp-server-subnet 3? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 3 deleted.
```

3 Delete the floating IP interfaces.

```
zSH> delete ip-interface-record f1t1/ip
ip-interface-record f1t1/ip
1 entry found.
Delete ip-interface-record f1t1/ip? [y]es, [n]o, [q]uit : y
ip-interface-record f1t1/ip deleted.
zSH> delete ip-interface-record f1t2/ip
ip-interface-record f1t2/ip
1 entry found.
```

```

Delete ip-interface-record flt2/ip? [y]es, [n]o, [q]uit : y
ip-interface-record flt2/ip deleted.
zSH> delete ip-interface-record flt3/ip
ip-interface-record flt3/ip
1 entry found.
Delete ip-interface-record flt3/ip? [y]es, [n]o, [q]uit : y
ip-interface-record flt3/ip deleted.

```

4 Delete the GPON traffic profiles.

```

zSH> delete gpon-traffic-profile 1
gpon-traffic-profile 1
1 entry found.
Delete gpon-traffic-profile 1? [y]es, [n]o, [q]uit : y
gpon-traffic-profile 1 deleted.
zSH> delete gpon-traffic-profile 2
gpon-traffic-profile 2
1 entry found.
Delete gpon-traffic-profile 2? [y]es, [n]o, [q]uit : y
gpon-traffic-profile 2 deleted.
zSH> delete gpon-traffic-profile 3
gpon-traffic-profile 3
1 entry found.
Delete gpon-traffic-profile 3? [y]es, [n]o, [q]uit : y
gpon-traffic-profile 3 deleted.

```

5 Delete the multicast control list.

```

zSH> delete mcast-control-entry 1/1
mcast-control-entry 1/1
1 entry found.
Delete mcast-control-entry 1/1? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/1 deleted.

```

6 Delete the video source.

```

zSH> delete video-source 1
video-source 1
1 entry found.
Delete video-source 1? [y]es, [n]o, [q]uit : y
video-source 1 deleted.

```

Static routes

This section discusses configuring static routes and includes:

- [Add routes, page168](#)
- [IP fallback route, page169](#)

Add routes

To add static routes, use the **route add** command. The command uses the following syntax:


```
route [source] add destination mask next-hop cost
```



Note: The word **default** can be substituted for a 0.0.0.0 destination and mask.

The following example creates a route to the subnet 192.178.21.0/24 using the gateway 192.172.16.1:

```
route add 192.178.21.0 255.255.255.0 192.178.16.1 1
```

The following example creates a default route using the gateway 192.172.16.1:

```
route add default 192.178.16.1 1
```

IP fallback route

The MXK supports IP redundancy or fallback IP routes. A fallback route is a second static route with the same destination and netmask of an existing route but with a different nexthop destination. The redundant or fallback route is used when the original nexthop destination is unavailable. The fallback route continues to be used until the revertive period expires. At that time, traffic switches back to the primary route.

A ping interval and ping retry count are used to determine route availability. The MXK pings the active nexthop router once during each ping interval. The ping-interval is specified in milliseconds and has a minimum value of 500 milliseconds or 1/2 second. If the number of ping failures to the current nexthop destination exceed the ping-fail-max setting, the current nexthop destination is replaced in the routing table with the fallback nexthop destination. The system begins pinging the new nexthop router and monitoring the number of ping failures. The revertive period is set by the system based on a multiple of the ping interval and retry count.



Note: The cost (metric) of the fallback route is automatically calculated to be one more than the cost of the first active route.

Configuring IP redundancy

Configure IP redundancy.

- 1 Add a route with the IP addresses of the nexthop router and fallback router.

```
zSH> route add 10.10.1.0 255.255.255.0 192.168.34.254 1 fallback
192.168.34.201 3000 5
```

- 2 Display the configured IP routes.

```
zSH> route show ...
Source Routing Table
```

Dest	Nexthop	Cost	Owner	Interface

Destination Routing Table

Dest	Nexthop	Cost	Owner	Fallback

0.0.0.0/0	192.168.34.254	1	STATICLOW	
10.10.1.0/24	192.168.34.254	1	STATIC	192.168.34.201
192.168.34.0/24	1/1/1/0/ip	1	LOCAL	

3 Delete the primary and fallback routes.

```
zSH> route delete 10.10.1.0 255.255.255.0 192.168.34.254 fallback
192.168.34.201
```

RIP configuration

RIP behavior for the system as a whole is configured in the **rip-global-config** profile. Each IP interface is then configured for RIP using the **rip** command. Currently, the MXK supports RIP v1 and v2. Note that the only routing domain currently supported is domain 1.

Configuring RIP global defaults

The following example configures RIP global behavior on the MXK:

- 1 Enable RIP for the system as a whole:

```
zSH> rip enable
```

- 2 List **ip-interface** records:

```
zsh> list ip-interface-record
ip-interface-record ethernet2/ip
ip-interface-record ethernet3/ip
2 entries found.
```

- 3 Pick the interface on which to configure with RIP. To enable receipt of RIP version 1 or version 2 advertisements on an interface, use the **rip** command and specify the interface and the type of advertisements to receive:

```
zSH> rip interface ethernet2 listen v1v2
```

- 4 To enable transmission of RIP advertisements on an interface:

```
a zSH> rip interface ethernet2 talk v2
```

or

```
b zSH> rip interface ethernet2 talk v1compat
```

IP administrative procedures

The following IP administrative procedures are supported on the MXK:

- [Modify profiles created by host/interface add commands, page172](#)
- [Display hosts, page172](#)
- [Display interfaces, page173](#)
- [Display routing information, page173](#)
- [Delete hosts, page174](#)
- [Delete interfaces, page174](#)
- [Delete routes, page174](#)
- [DHCP logging, page175](#)

- [IP statistics commands, page178](#)

Modify profiles created by host/interface add commands

After profiles have been created by the **host add** and **interface add** commands there are two methods of modifying the profiles:

- You can perform a **host delete** or **interface delete**, which deletes all associated profiles, then re-create those profiles with another **host add** or **interface add** command, specifying changes in the command line.
- You can modify the individual profiles which have been created by **host add** and **interface add** commands.

The **host add**, and **host delete** commands, <slot> and <port> may be replaced with brackets containing numbers in series and/or (dash-separated) ranges; <port> may be replaced with wildcard '*' for all ports on the card. Refer to the *CLI Reference Guide* for a complete description of the command options and syntax.

Display hosts

Enter the **host show** command to display information on existing hosts configured on the MXK. The command displays the IP address of the floating interface, the subnet group to which the host belongs, whether the host is dynamically or statically assigned, and if the host has been assigned an IP address, and the number of IP addresses allowed the host.

 **Note:** Entering **host show** without specifying an interface may display more information than is useful.

zSH> host show				
Rd/Address	Interface	Group	T	Host Address
1 10.107.8.254	1-13-1-0-eth	1	D	<unassigned>
			D	<unassigned>
			D	<unassigned>
			D	<unassigned>
			D	<unassigned>
1 10.107.8.254	1-13-2-0-eth	1	D	<unassigned>
			D	<unassigned>
			D	<unassigned>
			D	<unassigned>
			D	<unassigned>
1 10.107.8.254	1-13-3-0-eth	1	D	<unassigned>
			D	<unassigned>
			D	<unassigned>
			D	<unassigned>
			D	<unassigned>

Display interfaces

Issue the **interface show** command to display interfaces:



Note: Entering **interface show** without specifying the interface may display more information than is useful.

```
zSH> interface show
3 interfaces
Interface      Status  Rd/Address      Media/Dest Address  IfName
-----
1/a/2/0/ip     DOWN   1 192.169.1.14/24  00:01:47:1a:fe:65   ethernet2
1/a/4/0/ip     UP     1 172.16.7.49/24   00:01:47:1a:fe:67   ethernet4-7
1/a/6/0/ip     UP     1 172.16.160.49/24  00:01:47:1a:fe:64   ipobridge-160
-----
```

Display routing information

This section discusses:

- [Displaying the routing table, page173](#)
- [Displaying RIP information, page173](#)

Displaying the routing table

To display the routing table, use the **route show** command:

```
zSH> route show
Destination Routing Table
Dest          Nexthop          Cost    Owner          Fallback
-----
0.0.0.0/0     172.16.160.254  2       STATICLOW
172.16.160.0/24 1/a/1/0/ip      1       LOCAL
-----
```

Displaying RIP information

To display Routing Information Protocol (RIP) information, use the **rip show** command:

```
zSH> rip show
RIP Globals
-----
Route      Route      Route      Admin      Update
Domain     Changes    Queries    State       Time
-----
1          0          0          disabled    30
-----
RIP Interface Statistics
-----
IfName      Recv Bad    Recv Bad    Updates
            Packets     Routes     Sent To
-----
```

ethernet1	0	0	0
ethernet2-777	0	0	0
1-13-8-0-eth	0	0	0
1-13-6-0-eth	0	0	0
1-13-9-0-eth	0	0	0

RIP Interface Configuration

Route Domain	IP Address	Last Update	Version	Recv Bad Packets	Recv Bad Routes

Delete hosts

There are several ways to use **host delete** to delete IP interfaces associated with an *interface/type*.

Deleting hosts using IP address

host delete <ip address> deletes the static host IP interface.

```
zSH> host delete 1-13-1-0/eth ip 192.168.49.2
Deleting host for 1-13-1-0/eth
```

Deleting hosts using unused

host delete unused <number> deletes the designated number of unassigned floating IP slots that have not yet been assigned an IP address.

```
zSH> host delete 1-13-2-0/eth unused 4
Deleting host for 1-13-2-0/eth
```

Deleting hosts using all

host delete all deletes *all* of the hosts on this subnet and the subnet itself.

```
zSH> host delete 1-13-1-0/eth all
Deleting host for 1-13-1-0/eth
```

Delete interfaces

Issue the **interface delete** command to delete IP interfaces:

```
zSH> interface delete 1-13-6-0-eth/ip
Delete complete
```

Delete routes

To delete static routes, use the **route delete** command. The command uses the following syntax:

```
zSH> route delete destination mask next-hop
```

The following example deletes the network route to 192.178.21.0 using the gateway 192.172.16.1:

```
zSH> route delete 192.178.21.0 255.255.255.0 192.178.16.1
```

DHCP logging

This section covers:

- [Enable DHCP logging, page175](#)
- [DHCP server log messages, page176](#)
- [View client leases, page176](#)

Enable DHCP logging

The MXK provides a logging facility to monitor the DHCP packets it sends and receives. By default, DHCP messages are not displayed.

Enabling DHCP logging

- 1 Enable the DHCP server log messages:

```
zSH> log level dhcpserver info
Module: dhcpserver at level: info
```

- 2 Enable logging for the session:

```
zSH> log session on
Logging enabled.
```

As DHCP server messages are sent and received, they are displayed on the console.



Note: This setting does not persist across system reboots. You must re-enable DHCP logging after a MXK reboot.

- 3 These messages can be captured to a file using your terminal's capture facility, or sent to a syslog server. For example:

```
zSH> new syslog-destination 1
Please provide the following: [q]uit.
address: --> {0.0.0.0}: 192.200.42.5 syslog server IP address
port: ----> {514}:
facility: -> {local0}:
severity: -> {debug}: info
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

DHCP server log messages

When a device sends a DHCP server request to the MXK, a message similar to the following is logged:

```
AUG 13 12:20:48: info      : 1/1/1084: dhcpserver: DhcpServerTask: DHCPREQUEST for
155.57.1.21 from 00:b0:d0:98:92:3d via if496
```

This message indicates that a request for the address 155.57.1.21 was received by the device with the MAC address 00:b0:d0:98:92:3d. The request came in over the interface number 496.

To find what physical interface this corresponds to, use the **ifxlate** command:

```
zSH> ifxlate 496
ifIndex: -----> {496}
shelf: -----> {1}
slot: -----> {10}
port: -----> {48}
subport: -----> {0}
type: -----> {hds12}
adminstatus: -----> {up}
physical-flag: -----> {true}
iftype-extension: -> {none}
ifName: -----> {1-10-48-0}
```

The MXK sends the following message when it acknowledges the DHCP request packet.

```
AUG 13 12:20:48: info      : 1/1/1084: dhcpserver: DhcpServerTask: DHCPACK on
155.5 7.1.21 to 00:b0:d0:98:92:3d via if496
```

View client leases

Viewing client leases

When the MXK issues a DHCP client lease, it creates a **dhcp-server-lease**. You can view these records to see the status of the lease:

- 1 List the current leases.

```
zSH> list dhcp-server-lease
dhcp-server-lease 0/155/57/1/10
dhcp-server-lease 0/155/57/1/11
dhcp-server-lease 0/155/57/1/12
dhcp-server-lease 0/155/57/1/13
dhcp-server-lease 0/155/57/1/14
dhcp-server-lease 0/155/57/1/15
dhcp-server-lease 0/155/57/1/17
dhcp-server-lease 0/155/57/1/18
dhcp-server-lease 0/155/57/1/19
dhcp-server-lease 0/155/57/1/16
dhcp-server-lease 0/155/57/1/20
dhcp-server-lease 0/155/57/1/21
```



```

dhcp-server-lease 0/155/57/1/22
dhcp-server-lease 0/155/57/1/23
14 entries found.

```

2 View an individual record.

```

zSH> get dhcp-server-lease 0/155/57/1/10
starts: -----> {1060700857}
ends: -----> {1060700917}
flags: -----> {0}
hardware-address: --> {00:00:c5:90:3b:08}
client-identifier: -> {}
client-hostname: ---> {}
hostname: -----> {}
dns-fwd-name: -----> {}
dns-rev-name: -----> {}

```

Note that **0/155/57/1/10** represents routing domain 0, and the IP address 155.57.1.10.

IP statistics commands

The following IP commands are available to users with administrative privileges.

- **ip icmpstat**

Displays ICMP statistics.

```
zSH> ip icmpstat
```

ICMP:

```
0 call to icmp_error
0 error not generated because old message was icmp
0 message with bad code fields
0 message < minimum length
0 bad checksum
0 message with bad length
0 message response generated
```

- **ip ifstat**

Displays interface statistics.

```
zSH> ip ifstat
```

ifName	rxpkt	txpkt	rxmc	txmc	ierr	oerr	sqsz	sqdp
lo	19	19	0	0	0	0	0	0
ethernet1	860	63	832	2	0	0	0	0
1-13-2-0-eth	0	0	0	0	0	0	0	0
1-13-1-0-eth	0	0	0	0	0	0	0	0
ethernet2	1	1	0	0	0	0	0	0
5 interfaces								

- **ip ifsum**

Displays a summarized list of known interfaces.

```
zSH> ip ifsum
```

```
lo SOFTWARELOOPBACK ifindex 0 (ifp 0x315d558, 7|4)
  Flags: UP LOOPBACK MCAST ARP RUNNING
  inet 127.0.0.1 netmask 255.0.0.0
ethernet1 ETHERNETCSMACD ifindex 727 (ifp 0x31cb2a0, 9|3)
  Flags: UP BCAST MCAST IPFWD MCASTFWD ARP RUNNING CFGCURRENT
  inet 172.16.160.49 netmask 255.255.255.0 bcast 172.16.160.255
1-13-2-0-eth PROPVIRTUAL ifindex 723 (ifp 0x31cb6c0, 4|0)
  Flags: DOWN POINT-TO-POINT BCAST MCAST IPFWD MCASTFWD ARP RUNNING CFGCURRENT
UNNUMBERED
  inet 10.102.8.1 netmask 255.255.255.255 destinet 0.0.0.0
1-13-1-0-eth PROPVIRTUAL ifindex 720 (ifp 0x31cbae0, 4|0)
  Flags: DOWN POINT-TO-POINT BCAST MCAST IPFWD MCASTFWD ARP RUNNING CFGCURRENT
UNNUMBERED
  inet 10.101.8.1 netmask 255.255.255.255 destinet 0.0.0.0
ethernet2 ETHERNETCSMACD ifindex 733 (ifp 0x31cbf00, 7|1)
  Flags: DOWN BCAST MCAST IPFWD MCASTFWD ARP RUNNING CFGCURRENT
  inet 192.169.1.14 netmask 255.255.255.0 bcast 192.169.1.255
5 interfaces
```

- **ip inetstat**

Displays the active TCP/UDP/RAW endpoints terminating on the card.

```
zSH> ip inetstat
Active Internet connections (including servers)
PCB      Proto Recv-Q Send-Q Local Address      Foreign Address    (state)
-----
40dce5c  TCP      0      126  172.16.160.49.23   172.16.48.178.3326 ESTABLISHED
40de9b0  TCP      0      0   172.16.160.49.23   172.16.88.168.2819 ESTABLISHED
40a3cac  TCP      0      0   0.0.0.0.80         0.0.0.0.0         LISTEN
40a3a18  TCP      0      0   0.0.0.0.22         0.0.0.0.0         LISTEN
40a3994  TCP      0      0   0.0.0.0.23         0.0.0.0.0         LISTEN
40a3ebc  UDP      0      0   0.0.0.0.67         0.0.0.0.0
40a3e38  UDP      0      0   0.0.0.0.69         0.0.0.0.0
40a3c28  UDP      0      0   0.0.0.0.520        0.0.0.0.0
40a3ba4  UDP      0      0   0.0.0.0.162        0.0.0.0.0
40a3a9c  UDP      0      0   0.0.0.0.161        0.0.0.0.0
40a3910  UDP      0      0  127.0.0.1.1025     127.0.0.1.1024
40a388c  UDP      0      0   0.0.0.0.1024       0.0.0.0.0
40a3808  UDP      0      0   0.0.0.0.0          0.0.0.0.0
40a3f40  RAW      0      0   0.0.0.0.0          0.0.0.0.0
40a3db4  RAW     1208    0   0.0.0.0.0          0.0.0.0.0
40a3d30  RAW      0      0   0.0.0.0.0          0.0.0.0.0
40a3b20  RAW      0      0   0.0.0.0.0          0.0.0.0.0
```

- **ip ipstat**

Displays IP statistics.

```
zSH> ip ipstat
total 33058
badsum 0
tooshort 0
toosmall 0
badhlen 0
badlen 0
infragments 0
fragdropped 0
fragtimeout 0
forward 13939
cantforward 62
redirectsent 0
unknownprotocol 0
nobuffers 0
reassembled 0
outfragments 0
noroute 0
fastfwd 0
fastfwdnoroute 0
ffwdnointerface 0
nointerface 0
c2ctotal 0
c2cbadptr 0
c2cnopkt 0
```

c2cnoipktmem	0
c2ccorruptpkt	0
c2cttlexp	0
c2clastchance	0
flingnoipkt	0
flingerror	0
flung	0
rawflung	0
rawnofling	0
fwdloopdrop	0
localfastpath	31232
pendingarpoverflow	5

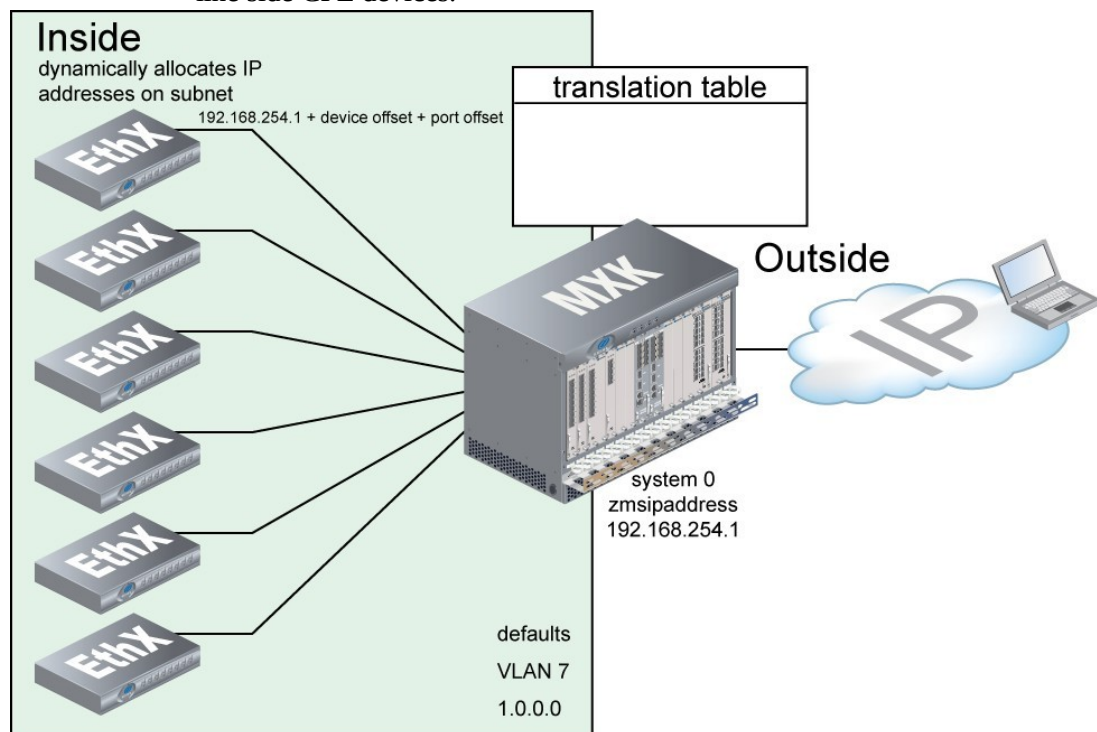
- **ip tcpstat**
Displays TCP statistics.
- **ip udpstat**
Displays UDP statistics.
- **ip arpdelete**
Deletes an entry from the ARP table.
- **ip arpflush**
Flushes the ARP table of all entries.
- **ip arpshow**
Displays the ARP table.

CPE Manager

The MXK's CPE Manager provides a means for managing consumer premises equipment (CPE) devices without requiring extra routable IP addresses to reach these CPE end-points. While the CPE Manager is specifically designed for Zhone's EtherXtend and Active Ethernet zNID family of CPE products, CPE Manager can be used with any CPE device which supports IP addresses on a VLAN.

In many service provider networks, the increasing usage of IP-aware CPE devices creates an operational challenge for service providers because the number of devices which require IP addresses cause IP address space depletion, making it hard to assign routable addresses for these devices.

A solution to this problem is the SLMS CPE Manager. CPE Manager adds proxy capability to SLMS, allowing one IP interface on the Zhone central office device to provide IP access to all the subtended CPE devices connected to it. This one IP interface is created on an upstream port which is routable on the service providers management network, and it provides IP address and protocol port translation when forwarding packets to and from managed CPE devices. In this way, IP can be used for CPE management without having to consume IP address space or having to add network routes for reachability of line side CPE devices.



To access a CPE configured using CPE Manager, access the MXK through its IP address, however, instead of using the well known protocol ports, use the CPE's base public port plus an offset to the specific port used for the protocol desired. Supported protocols include Echo, FTP (data), FTP (control), SSH,

Telnet, HTTP, SNMP and HTTPS. A list of offsets for public ports is given in [Offsets for public ports, page182](#).

CPE Manager is supported on the following line cards:

- MXK-EFM-SHDSL-24-NTWC
- MXK-EFM-SHDSL-24-NTP
- MXK-AEX20-FE/GE-2S
- MXK-AEX20-FE/GE



Note: Up to 480 CPE devices may be managed using the CPE manager from a single MXK at this time.

Table 24: Offsets for public ports

Well known port	Type	Name	Public port offset
7	TCP, UDP	ECHO	+0
20	TCP	FTP - data	+1
21	TCP	FTP - control	+2
22	TCP, UDP	SSH	+3
23	TCP, UDP	Telnet	+4
80	TCP	HTTP	+5
81	TCP	HTTP	+6
161	TCP, UDP	SNMP	+7
443	TCP	HTTPS	+8

The private class A network is set up by default as 1.0.0.0/8 on VLAN 7. These defaults may be changed, see [Changing the VLAN or class A network used as the CPE manager local network, page183](#).

The IP addresses given to CPEs follow the general guidelines:

<Class A network>.<Slot>.<Port number: higher order byte>.<Port number: lower order byte>

By default we use the 1.0.0.0 class A network. In other words, a class A network is one that has an 8 bit mask which means only the first byte of the IP address is common between nodes in the network. If you execute the following command: **cpe-mgr add local network 2.0.0.0**, the class A network will be changed and all local IP will start with 2.

Configuring the MXK as a CPE manager for Active Ethernet

Setting up the CPE manager from the CLI is fairly simple. First you have to have an IP address on an upstream port.

- 1 Add a public address for the CPE manager

```
zSH> cpe-mgr add public 192.168.254.1
CPE Manager using 192.168.254.1 for public interface.
```

Adding the public address for the MXK requires that the MXK has already been given an IP address.

- 2 Add the local device to the CPE manager.

```
zSH> cpe-mgr add local 1-13-1-0/eth

Configured CPE Manager's local network:
  Class A network: 1.0.0.0
  Local IP:       1.0.0.1
  VLAN ID:       7
Created CPE Management interface: 1-13-1-0-eth-7/ip
```

Configuring the MXK as a CPE manager for EFM-SHDSL

To create an EFM-SHDSL bond group, see [Bond group configuration, page 655](#).

- 1 Add a public address for the CPE manager

```
cpe-mgr add public 192.168.254.1
```

Adding the public address for the MXK requires that the MXK has already been given an IP address.

- 2 Add the local device to the CPE manager.

```
cpe-mgr add local 1-3-42-0/efmbond
```

Changing the VLAN or class A network used as the CPE manager local network

Ordinarily the default settings are acceptable. However if you need to change the default class A network or VLAN ID you can use the following commands:

- 1 To change the VLAN ID for the CPE manager local private network

```
cpe-mgr add local vlan <vlan id to use internally for management>
```

If you were to manually set the VLAN ID to the default, you would use

```
cpe-mgr add local vlan 7
```

- 2 To manually set the local network settings

```
cpe-mgr add local network <class A network used internally for all managed CPEs>
```

If you were to manually set the local network to the default, you would use

```
cpe-mgr add local network 1.0.0.0
```



Note: You can only manually set the local network settings when no CPE devices are currently configured on the network.

Verifying CPE Manager

To verify or troubleshoot CPE manager, you should understand what the two commands for CPE manager do. The first **cpe-mgr add public** command

- Sets **natenabled** to “yes” in the **ip-interface-record** for the public address (in our example, the 192.168.254.1 address)

The second command, **cpe-mgr add local**:

- Creates a floating **ip-interface** record with IP address of 1.0.0.1
- Creates an **ip-unnumbered-record** for the floating ip-interface record
- Creates a **dhcp-server-subnet** for the 1.0.0.0 network
- Creates a host **ip-interface-record** for the CPE on interface (in our example bond group)

Assigns a local IP address based on the interface description (not routable, but may be reached from the private local network, or by Telnet to the MXK, then Telnet from the MXK to the device)

- Creates a **pat-bind** profile of type CPE manager



Note: The **ip-interface-record** created is not a normal “host” record and cannot be seen using the **host show** command.

The pat-bind profile for the first device from the example contains the local IP address (1.3.0.42) and the CPE base port (51921):

```
zSH> get pat-bind 1
pat-bind 1
public-ipaddr: -> {192.168.254.1}
public-port: ---> {51921}
local-ipaddr: --> {1.3.0.42}
local-port: ----> {9}
portType: -----> {cpemgr}
```

The local address which is given is based on the interface in the form:

```
<local class A network>.<slot>.<port HI byte>.<port LO byte>
```

From our example bond group, *1-3-42-0/efmbond*, the local IP address (as shown above in the **pat-bind 1** profile) is 1.3.0.42. If you need to verify this number, do a get on the **pat-bind** profile.

Additional information about CPE manager

The first device will be accessible by the MXK's public IP address and the CPE base port. The CPE base port for the first device is 51921. To reach one of the well known ports you then give the offset for the public port. Well known port (7) is for echo which has an offset of zero.

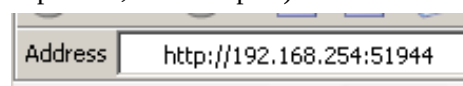
1st device	ECHO	+0	51921
	FTP (data)	+1	
	FTP (control)	+2	
	SSH	+3	
	Telnet	+4	
	HTTP	+5	
	HTTP	+6	
	SNMP	+7	
	HTTPS	+8	
2nd device	ECHO	+0	51930
	FTP (data)	+1	
	FTP (control)	+2	
	SSH	+3	
	Telnet	+4	
	HTTP	+5	
	HTTP	+6	
	SNMP	+7	
	HTTPS	+8	
3rd device	ECHO	+0	51938
	FTP (data)	+1	
	FTP (control)	+2	
	SSH	+3	
	Telnet	+4	
	HTTP	+5	
	HTTP	+6	
	SNMP	+7	
	HTTPS	+8	

To telnet to the first CPE via the well known port, 23, you would use the CPE base port plus the public port offset of 4; You would use the MALC's address (192.168.254.1), then 51925 (51921 + 4) to Telnet to the device. From a Unix or DOS prompt it would look like

```
telnet 192.168.254.1 51925
```

To access the second device you need to start with the CPE base port for that device. Each device consumes nine public ports, so the first device has a port range from 51921 - 51929, the second device has a port range from 51930 - 51938, the third from 51939 - 51947 and so on.

To access the HTTP port on the third device from a browser, you would start from the first public port address 51921 + 18 (the 51921 start point plus two times nine for the first two devices to get to the third device range) + 5 (to get to port 80, a HTTP port) or 51944.



As CPE devices are deleted or added, holes will form in the list of CPE devices, so the order eventually becomes arbitrary, but is used in the discussion to elucidate how the mechanism works. To find the CPE base port you can do an **interface show**, then **get pat-bind *** to find the CPE device you want. The pat-bind profile's public-port is the CPE base port for the device.

Finding the local IP address and CPE base port

To find the local port to access a CPE device use the **interface show** command to find the IP address and **get pat-bind *** to find the CPE base port.

```
zSH> interface show
```

```
5 interfaces
```

Interface	Status	Rd/Address	Media/Dest	Address	IfName
1/1/1/0/ip	UP	1	192.168.254.108/24	00:01:47:05:9c:bd	ethernet1-1
1/3/42/0/ip	UP	1	2.2.2.1/24		1-3-42-0-efmbond
1/3/42/0/ip	UP	1	[1.0.0.1]	1.3.0.42	1-3-42-0-efmbond-7
1/3/45/0/ip	UP	1	[1.0.0.1]	1.3.0.45	1-3-45-0-efmbond-7
1/3/208/0/ip	UP	1	[1.0.0.1]	1.3.0.208	1-3-208-0-n2nbond-7

```
-----
1/1/1/0/ip  UP      1 192.168.254.108/24 00:01:47:05:9c:bd ethernet1-1
1/3/42/0/ip  UP      1 2.2.2.1/24                               1-3-42-0-efmbond
1/3/42/0/ip  UP      1 [1.0.0.1]           1.3.0.42           1-3-42-0-efmbond-7
1/3/45/0/ip  UP      1 [1.0.0.1]           1.3.0.45           1-3-45-0-efmbond-7
1/3/208/0/ip  UP      1 [1.0.0.1]           1.3.0.208          1-3-208-0-n2nbond-7
-----
```

```
zSH> get pat-bind *
```

```
pat-bind 4
```

```
public-ipaddr: -> {192.168.254.108}
```

```
public-port: ---> {51948}
```

```
local-ipaddr: --> {1.3.0.42}
```

```
local-port: ----> {9}
```

```
portType: -----> {cpemgr}
```

```
pat-bind 2
```

```
public-ipaddr: -> {192.168.254.108}
```

```
public-port: ---> {51930}
```

```
local-ipaddr: --> {1.3.0.208}
```

```
local-port: ----> {9}
```

```
portType: -----> {cpemgr}
```

```
pat-bind 1
```

```
public-ipaddr: -> {192.168.254.108}
```

```
public-port: ---> {51921}
```

```
local-ipaddr: --> {1.3.0.45}
```

```
local-port: ----> {9}
```

```
portType: -----> {cpemgr}
```

Web UI cut-through for EtherXtend devices

This section provides the configuration procedure to create hyperlinks in the MXK Web UI that when clicked, will take you to the Web UI for the EtherXtend 3400. See [Figure15](#) and [Figure16](#).

Creating a Web UI cut-through for EtherXtend devices

From the MXK CLI:

- 1 Create a management interface for the MXK.
- 2 Create a CPE public IP using the MXK management IP.

```
zSH> cpe-mgr add public 172.24.200.163
CPE Manager using 172.24.200.163 for public interface.
```

- 3 Create an EFM bond group, then add the links.

```
zSH> bond add group 1-1-25-0/efmbond
Group ID {25} is already in use.
Bond group - bond-0032/efmbond - was successfully created.
```

```
zSH> bond add member bond-0032/efmbond 1-1-1-0/shdsl
```

```
zSH> bond add member bond-0032/efmbond 1-1-2-0/shdsl
```

```
zSH> bond add member bond-0032/efmbond 1-1-3-0/shdsl
```

```
zSH> bond add member bond-0032/efmbond 1-1-4-0/shdsl
```

- 4 Create a local **cpe-mgr** IP for the bond group.

```
zSH> cpe-mgr add local bond-0032/efmbond
Created CPE Management interface: bond-0032-efmbond-7/ip
```

- 5 View the **pat-bind** record that was automatically created.

```
zSH> get pat-bind *
public-port: ---> {51930}
local-ipaddr: --> {1.1.0.32}
local-port: ----> {9}
portType: -----> {cpemgr}
pat-bind 1
public-ipaddr: -> {172.24.200.163}
```

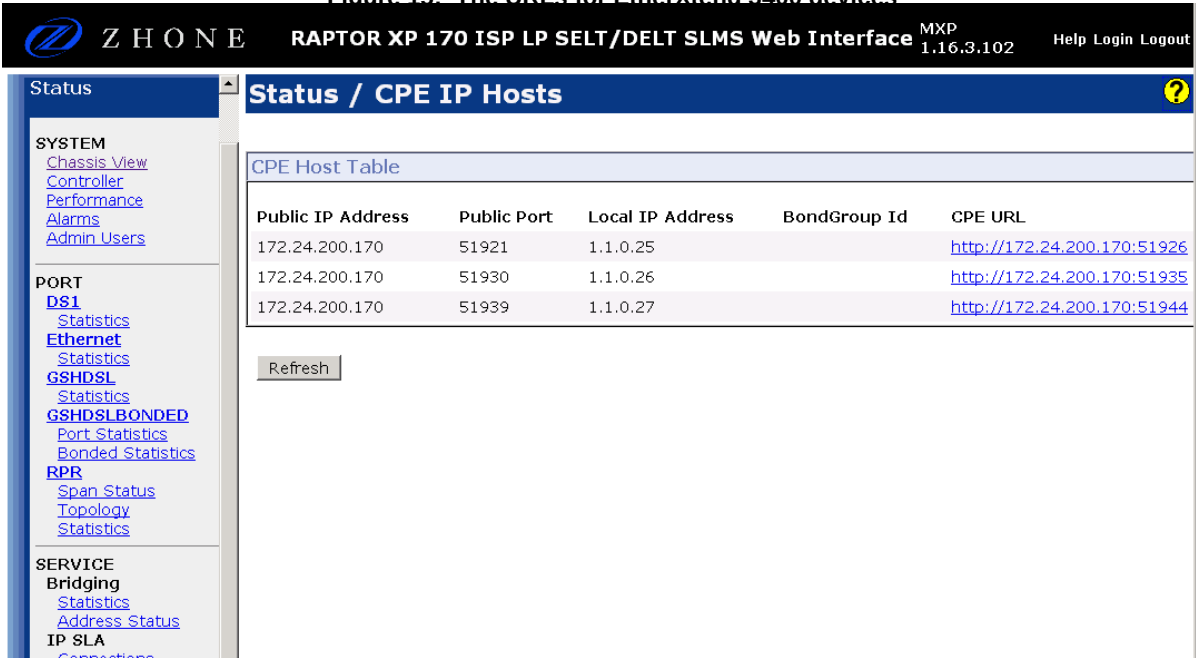
- 6 Verify the bond group **cpe-mgr** IP interface is UP.

```
zSH> interface show
```

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/1/1/0/ip	UP	1 172.24.200.163/24	00:01:47:1a:db:0f	ethernet1-1-200
1/1/32/0/ip	UP	1 [1.0.0.1] 1.1.0.32		bond-0032-efmbond-7

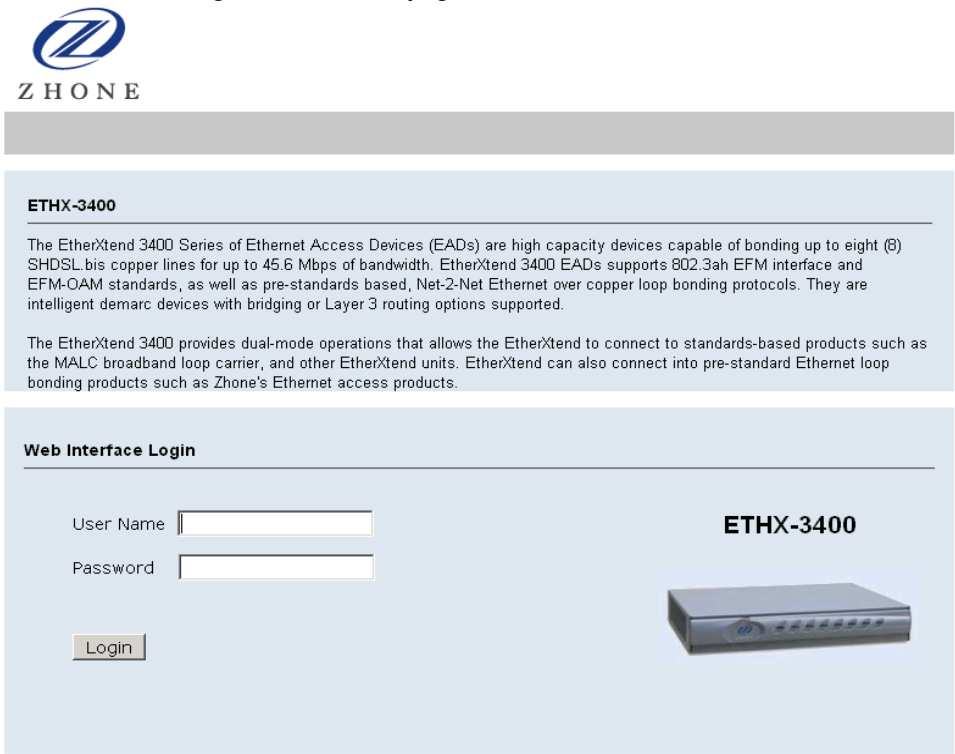
- 7 From a browser, launch a Web UI to the MXK management interface, *http://172.24.200.163*.
- 8 Through the WebUI, view the CPE Cut-Through URL by clicking to *Status->Service->CPE->CPE IP Hosts*.

Figure 15: The URLs for EtherXtend 3400 devices



- 9 Click on the CPE URL to launch the WebUI for the EtherXtend 3400.

Figure 16: Web UI page for the ExtherXtend 3400



5 BRIDGING CONFIGURATION

This chapter covers Zhone's bridging services including:

- [Overview, page190](#)
- [Terminology and concepts, page191](#)
- [SLMS bridge types, page198](#)
- [Tagging operations, page211](#)
- [Common bridge commands, page226](#)
- [Settings for asymmetric bridges, page229](#)
- [Settings for symmetric bridges, page230](#)
- [Bridge configuration with DHCP relay, page232](#)
- [Shaping Traffic: Class of Service Queuing, page235](#)
- [PPPoA - PPPoE interworking, page237](#)
- [Bridges with packet rule records, page240](#)
- [Advanced bridging topics, page260](#)
- [MXK bridging configurations, page274](#)
- [Administrative commands, page310](#)

Zhone SLMS bridging services provide a comprehensive suite of capabilities. Zhone's SLMS software and its three management interfaces — CLI, ZMS, and Web UI — provide an extremely flexible mechanism for defining bridges. Combine the SLMS software with Zhone's Multi-Service Access Platforms (MXK and MALC) and line of 1U MSAPs/DSLAMs (such as the MALC XP, hardened Raptor XP) and CPEs such as the EtherXtend) to form a myriad of Zhone solutions which can support any network design.

Zhone's MSAPs blend of high performance, high density integrated access platforms allow providers the flexibility to match the network to the needs of their markets. The fixed 1U MSAPs/DSLAMs carry high performance access closer to business and residential consumers from Central Office style deployments to remote cabinets and Multi-Dwelling units. The ability to combine FTTx, and copper solutions such as EFM-SHDSL and ADSL2+ completes a very effective access picture.

Bridging services are primarily configured through the use of the **bridge add** command. The **bridge add** command creates a logical interface specifying

the parameters for the bridge interface (bridge type, VLAN ID, tagging, COS options, and other parameters). This logical interface is stacked on a physical interface like an Ethernet, ADSL or GPON interface.

This chapter describes fundamental concepts necessary to understand how to build bridges using the MXK and other Zhone SLMS based devices.

The bridging fundamentals described in this chapter do not intend to cover logical link layer bridging in an in depth or comprehensive manner, but are provided to highlight Zhone's mechanisms for providing bridging services.

Overview

Whether discussing bridging or routing, the main function of SLMS MSAPs and DSLAMs is to forward packets (IP) or frames (bridging).

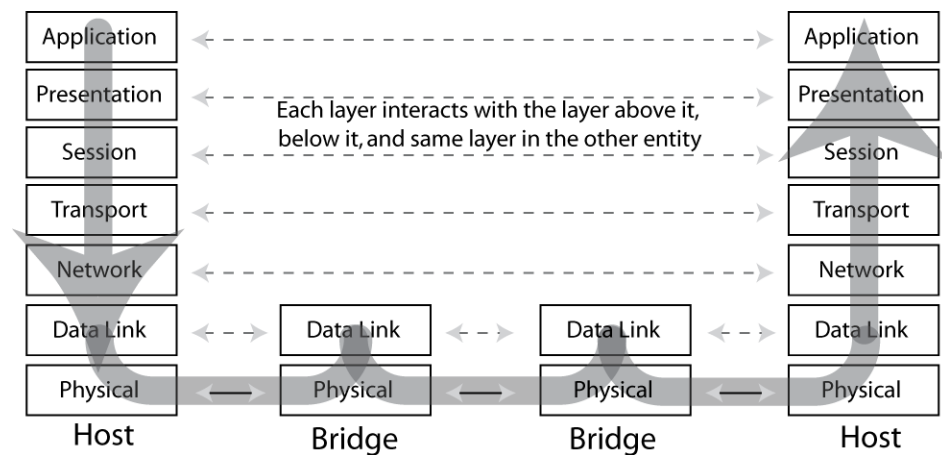
- Frames are delivered on MAC address (ISO Logical Link layer 2, bridging)
- Packets are delivered based on IP address (ISO Network layer 3, routing)

The layers referred to above are part of the Open Systems Interconnection (OSI) reference model. While not all protocols follow the OSI model, the OSI model is helpful for understanding variations of network functionality.

Table 1: ISO Open Systems Interconnection Reference Model

Layer	Name	Function	
7. Application		Network processes and application interactions	Host Layers
6.	Presentation	Mapping between application and lower layers — data presentation and encryption	
5.	Session	Manages connections between local and remote application.	
4.	Transport	Manages the end to end connection, reliability, tracks segments and retransmission (error control)	
3.	Network	Routing functions. Transferring data from source to destination. The best known layer 3 protocol is Internet Protocol (IP).	Media Layers
2.	Data Link	Transfers data between network entities.	
1.	Physical	Relationship between the transport medium (copper, fiber, wireless) and devices	

If an application on one host requests information from another networked application on another host (for example clicking a link to another page in a browser), the requests proceed down the layers until it is transmitted on the physical media (wire, fiber, wireless signal), until the message is picked up at the other end and progresses up the layers as shown in [Figure1](#). The response follows the same process.

Figure 1: Applications requested networked information

Bridges direct frames based on address information in the frame as well as information learned from the processing and directing of other frames. The processing and directing of frames is the learning, forwarding, or filtering that is done by the device. The amount of processing and information read from the frame is kept to a minimum to enhance the throughput speed of the device.

Terminology and concepts

This section covers:

- [Physical port, page192](#)
- [Physical interface, page192](#)
- [Logical interface, page193](#)
- [Bridges and bridge interfaces, page193](#)
- [VLANs and SLANs, untagged, tagged and stagged, page193](#)
- [Upstream and downstream, page196](#)
- [Broadcast, multicast, and unicast, page198](#)

Zhone SLMS bridging services draw from many specifications to provide a comprehensive suite of capabilities — IEEE 802.1-2004 (basic L2 bridging capabilities), IEEE 802.1W (Rapid Spanning Tree), DSL-Forum TR-101 and TR-156 (Ethernet backhaul for access devices, VLAN capabilities). Often there is not one specification to draw a set of terminology. Zhone uses a combination of terms from accepted standards, specifications, or Zhone's own terminology where no clear industry accepted term exists.

It is important to understand how the physical relates to the conceptual in building networks.

Physical port

The physical port is the physical connection on a device, essentially the layer 1 physical port. Examples of physical ports include

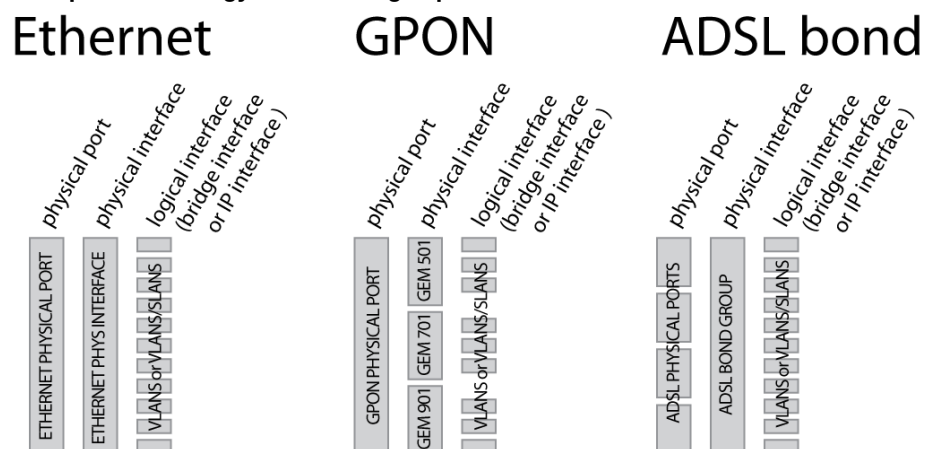
- Ethernet physical medium (Fast Ethernet or Gigabit Ethernet)
- Individual wire pair for POTs or xDSL
- GPON OLT port

The physical port is not necessarily the physical connector. A Champ connector may have 50 individual wire pairs. The physical port in this case, is the individual wire pair. The physical port in GPON would be one fiber connection, however that one connection may be and usually will be shared with multiple subscriber devices.

Physical interface

A physical interface is all of, a subset of, or a collection of, physical ports depending on the capabilities of the transportation technology as shown in [Figure2](#).

Figure 2: Physical port to physical interface to logical interface vary by transport technology and bonding capabilities



The mapping of physical ports to physical interfaces may be:

- All of a physical port. With Ethernet, the physical interface is all of the physical port.
- A subset of a physical port. With GPON, GEM ports are used to separate a single physical port into multiple virtual ports.
- A collection of physical ports. Bonded links or IMA groups combine multiple physical ports into one logical interface.

Logical interfaces are associated with physical interfaces.

Logical interface

There are two types of logical interfaces — bridge interfaces and IP interfaces. These interfaces may be associated with all or part of the traffic on a physical interface. When the logical interface is broken down into connections, these connections are identified by a Virtual Local Area Network (VLAN) identifier, an ATM Virtual Connection (for connection based technologies such as ADSL), or both.

For information about IP interfaces, see [Configuring IP on page291](#).

Bridges and bridge interfaces

A bridge is a collection of bridge interfaces which share a common attribute to form a community of interest. The attribute which defines the community of interest is either a VLAN ID or a combination of SLAN ID and VLAN ID.

Frames received on an interface belonging to a bridge can only be sent to other interfaces in the system belonging to the same bridge. Many bridges can exist in the system at the same time, each one defined by the VLAN ID or SLAN ID/VLAN ID.

Bridges connect network segments. The ends of the bridge are the bridge interfaces as defined in the **bridge-interface-record** profile.

Unlike a repeater which has two interfaces and takes in data on one interface and pushes it out the other (normally to strengthen the signal) or a hub which has more than two interfaces and takes in data on one interface and pushes it out on all the other data interfaces — bridges are more complex. Bridges analyze the incoming data frames to determine where to forward each frame. Where the data comes in (ingress) and where the data goes out (egress) on the device are determined by the bridge configuration. Zhone primarily uses two types of bridges — Transparent LAN Services (TLS) bridges (which are called symmetric because all the bridge interfaces have the same behavior) and asymmetric bridges which can be broken down into three different bridge interface types, each with its own behavior. See [SLMS bridge types on page198](#).

Frames which ingress on one bridge interface are not forwarded back out that same bridge interface.

VLANs and SLANs, untagged, tagged and stagged

VLANs and SLANs are used to separate traffic. VLANs and SLANs are typically used to separate services such as in triple play scenarios (voice, video, and data). Voice and video services are provided from servers on private networks. The messages from the voice and video servers are similar and have the same priority, only the content is different. Data services come from a gateway to the public Internet and the content is not as similar as the voice or video.

VLANs separate the traffic of all services, so the known traffic is separated from the unknown traffic. This separation also provides the means for handling traffic differently through the use of Quality of Service (QoS) markings to prioritize voice and video traffic.

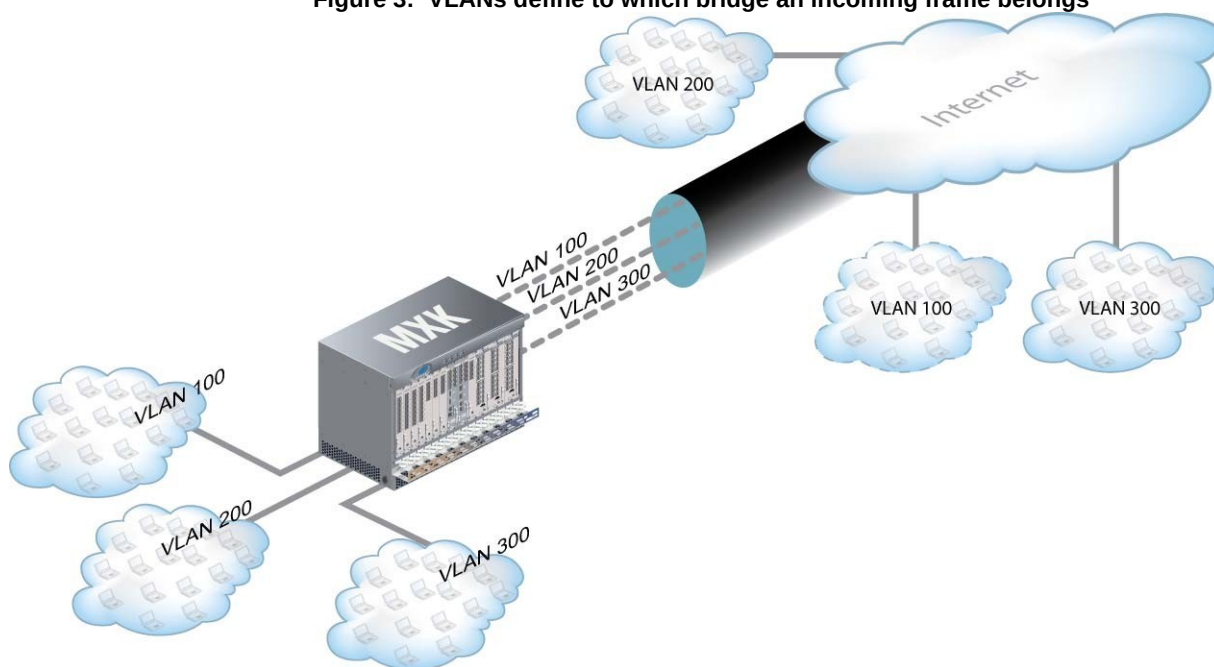
The separation of traffic allows for other mechanisms such as:

- conforming traffic to policies as with bandwidth limiting
For details see [Bandwidth limiting by port and service on page245](#)
- providing port-to-port security of users sharing a VLAN as with Destination MAC swapping.
For details see [Destination MAC swapping on page243](#)
- inserting identification information for DHCP servers
For details see [DHCP on bridge packet rules \(DHCP relay, Option 82, and Forbid OUI\) on page253](#)
- inserting tags for identification purposes as when the MXK is a PPPoE intermediate agent
For details see [PPPoE with intermediate agent on page256](#)

Another example of VLANs and SLANs is the separation of traffic for groups of hosts/users.

VLANs (and SLANs) may also be used for identifying the origination of frames as shown in [Figure3](#). See [Tagging operations](#) for some network design scenarios.

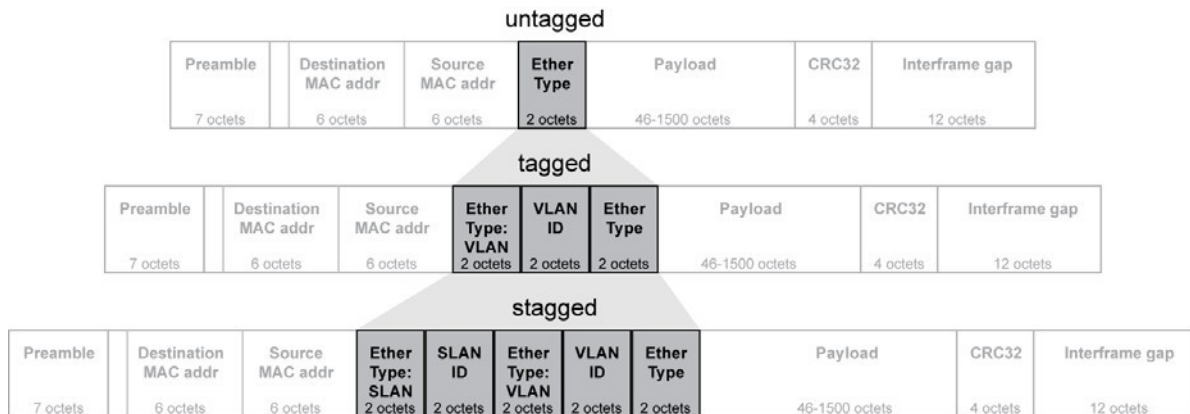
Figure 3: VLANs define to which bridge an incoming frame belongs



IEEE 802.1 Q-in-Q expanded the VLAN space in the Ethernet frame to support tagging already tagged frames. This second tag, an SLAN, creates a double-tagged Ethernet frame.

A frame which has no VLAN ID is referred to in the CLI as *untagged*. A frame which has a VLAN ID, but not an SLAN ID is single tagged, referred to as *tagged*. A frame which has both a VLAN ID and SLAN ID is double tagged, or *staged* as shown in Figure4.

Figure 4: Ethernet frames: untagged, single tagged and double tagged



Note: The octets for VLAN ID and SLAN ID include CoS information

Zhone's SLMS CLI uses a very flexible mechanism for defining bridge interfaces. When adding a bridge interface you can define the bridge interface to accept and send out untagged, tagged or staged frames. No other frames will be accepted. If a bridge interface is expecting a tagged frame (using the **bridge add** command with the *tagged* key word to create the bridge interface), then untagged frames or double tagged frames will not be handled by this bridge interface. If a double tagged frame is expected, untagged and single tagged frames will not be handled by this interface. Those frames may be handled by other bridge interfaces depending on the configuration.

Only one untagged bridge interface can exist on a port or sub-port since frames will not have a VLAN ID to match multiple bridge interfaces. Untagged bridges are created using the **bridge add** command with either the *untagged* key word or not using the key words to define single tagged (*tagged*) or double tagged (*staged*).

You can issue a **bridge add** command without specifying whether the bridge interface is to be *untagged*, *tagged* or *staged*. If you do not designate a tagging option, the bridge interface assigns a default tagging based on the type of bridge interface:

- downlink
untagged

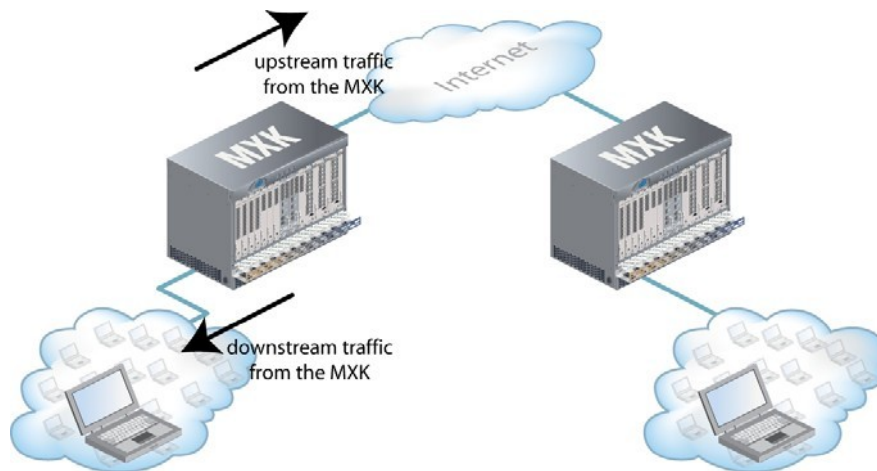
- uplink, intralink
tagged
- TLS
untagged
- wire
tagged In this case, must designate a VLAN or SLAN.

See [Tagging operations on page211](#) for more information on untagged, tagged, and staged traffic.

Upstream and downstream

Upstream and downstream are situational terms and are used in an SLMS device-centric manner. Typically the term upstream means the SLMS device's physical interface(s) are facing toward the core of the network and the term downstream means the device's physical interfaces is facing toward subscribers as described in [Figure5](#).

Figure 5: Upstream and downstream using the typical model



This model assumes a hierarchy, but neglects the notion that at some point the data stream must change from upstream to downstream (since it is going from one application to another, one host to another, one user to another, even if one of the applications is a video server. To the server company, the data stream is going upstream to the core to get to the client). In other words, there is no way of defining “up” clearly throughout the entire conceptual model. Therefore the terms upstream and downstream are used with the general understanding that upstream is toward the Internet core and downstream is toward the subscriber.

The terms upstream and downstream are closely associated with the bridge interface types *uplink* and *downlink*. Uplinks and downlinks have different specific behaviors which define the bridges.

The terms upstream and downstream are also used when discussing TLS interfaces. TLS interfaces have the same behavior for both upstream and downstream interfaces which may be advantageous for certain access situations.

Broadcast, multicast, and unicast

The purpose of a bridge is to transmit frames. In general, frames are received on one interface and then are transmitted out on one or more other interfaces. There are three general ways to transmit frames:

- unicast

Unicast frames are sent to a specific address.

- multicast

Multicast frames are sent to a limited number of entities.

- broadcast

Broadcasts are sent to all available entities, usually all devices in a subnet as they can be a reasonably limited set of entities.

Learning on bridge interfaces means that the interface learns the source MAC address from the Ethernet frame of a received frame and the MAC address (as well as the VLAN and bridge interface upon which the MAC address was received) is put in the forwarding database. See source and destination addresses in [Figure4](#) to see the structure of the Ethernet frame. When the learned MAC address from a previously received frame is the destination MAC address in an Ethernet frame the device forward the frame to the appropriate egress bridge interface.

There is no learning when receiving broadcast or multicast frames.

Each bridge type has a different behavior for learning the source address and forwarding to the destination of the received frame. The different behaviors in learning and forwarding are discussed in the following sections — TLS bridges and asymmetric bridges. The behavior of each bridge type with relation to the learning and forwarding behavior of unicast frames is also discussed in [SLMS bridge types](#).

SLMS bridge types

Zhone's SLMS devices, such as the MXK, MALC, MALC XP, Raptor XP, and SLMS based EtherXtend, use two types of bridges — symmetric bridges which have the same bridging behavior and asymmetric bridge which have different bridging behavior. The bridge interfaces for symmetric bridges provide the same bridging behavior (bridge interfaces for TLS are the one example of a symmetric bridge interface) and bridge interfaces for asymmetric bridges provide different bridging behavior. Uplink and downlink bridge configurations are the most common asymmetric bridges but intralink bridges are also asymmetric bridges. The different behavior for these four bridge types are useful in creating network bridges.

Symmetric bridges use TLS and wire bridge interfaces:

- TLS bridge interfaces have the same behavior regardless of which ports are being bridged.

A TLS bridge interface is created with a **bridge add** command and *tls* keyword.

TLS bridge interfaces only work in conjunction with other TLS bridge interfaces.

- Wire bridge interfaces, which are a reserved TLS bridge, have the same behavior regardless of the ports being bridged.

A wire bridge interface is created with the **bridge add** command and *wire* keyword.

A wire bridge is only connected to another wire bridge in a two bridge interface configuration and reserves a VLAN ID for two ports for the entire system.



Note: When a VLAN ID is used for two *wire* bridges, that VLAN ID cannot be used anywhere else on the MXK system.

Asymmetric bridges use three different bridge interface types:

- Uplinks

Uplinks are normally used for upstream traffic toward the Internet core.

An uplink bridge interface is created with a **bridge add** command and *uplink* keyword and a **bridge-path add** command for that uplink.

Uplink bridge interfaces only work in conjunction with asymmetric bridge interfaces.

- Downlinks

Downlinks are normally used for downstream traffic toward the subscribers.

A downlink bridge interface is created with a **bridge add** command and *downlink* keyword.

Downlink bridge interfaces only work in conjunction with asymmetric bridge interfaces.

- Intralinks

Intralinks are normally used for subtending other SLMS devices.

An intralink bridge interface is created with a **bridge add** command and *intralink* keyword and a **bridge-path add** command with keyword *intralink* for that intralink.

Intralink bridge interfaces only work in conjunction with asymmetric bridge interfaces.

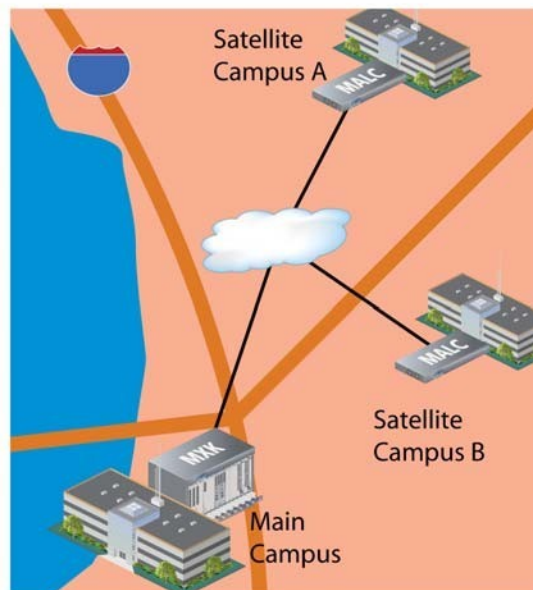
For more information see [Transparent LAN services](#) and [Asymmetric bridges](#).

Transparent LAN services

Transparent LAN services (TLS) bridges are used when you want traffic freely flowing among a community of users.

For example, a school district may use TLS bridges to extend a LAN to multiple campuses. The remote campuses will appear to be on the same LAN segment even though they are geographically separated.

Figure 7-10 TLS bridges connect multiple campuses as if one LAN

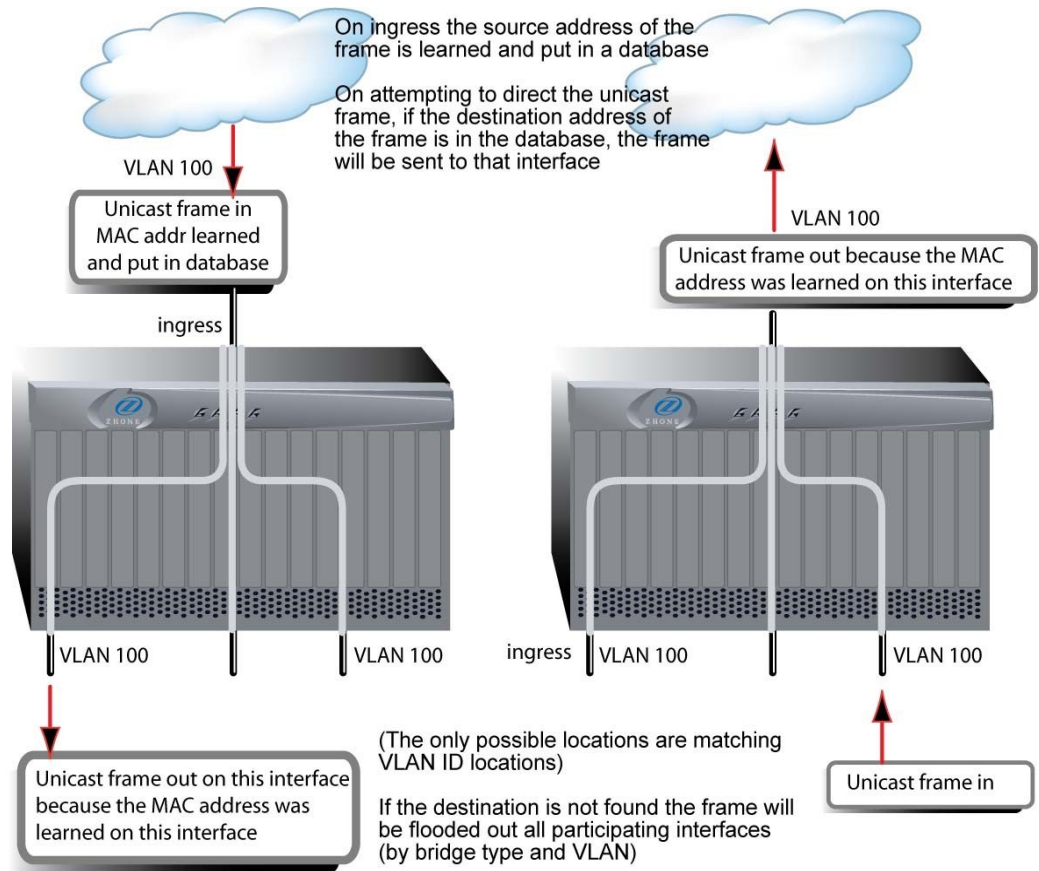


Another situation where TLS bridges are a good solution is for voice applications. The forwarding database does not retain information forever. Like all bridges, if there is no activity on the VoIP bridge, then the MAC address of the VoIP supplying access device will eventually time-out the MAC address of the VoIP in the bridge forwarding table. Upstream is the VoIP server which will try to send frames to the end VoIP supplying device. If no MAC address is in the forwarding table, the different type of bridges will behave differently. The TLS bridge will flood all the bridge interfaces of the TLS VoIP VLAN and rediscover the VoIP supplying access device. The downlink of an asymmetric bridge will discard the frame, so the call will not be completed.

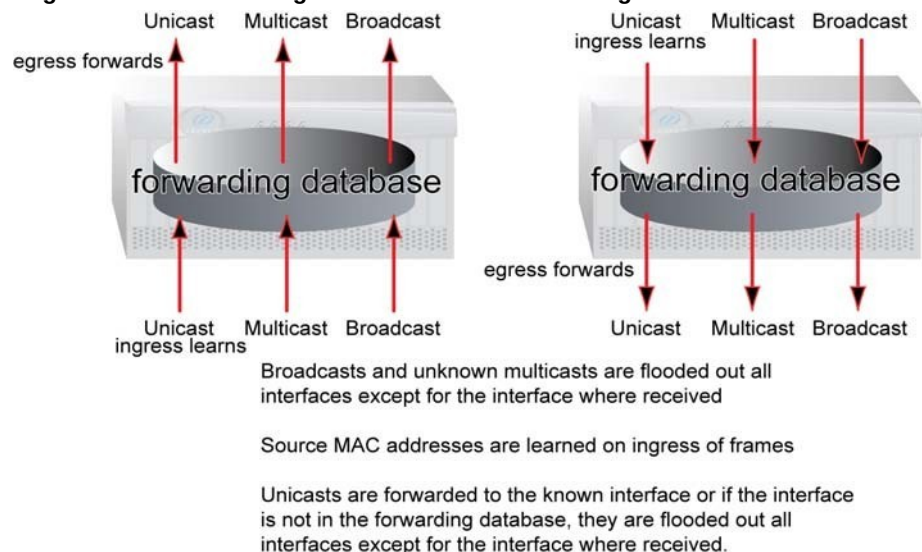
A TLS bridge interface is used only with other TLS bridge interfaces. TLS bridge interfaces are not used with any asymmetrical bridge interfaces.

All interfaces in a TLS bridge are treated the same as shown in [Figure 7](#). There is no designation of an uplink or a downlink. When describing the equal interfaces of a TLS bridge it is helpful to think in terms of ingress or egress on an interface.

TLS bridges learn MAC addresses of unicast frames and forward the frames to learned destinations. Broadcasts and unknown unicasts are flooded out all interfaces except the ingress interface.

Figure 7: In a TLS bridge all interfaces learn & forward the same

Frames entering the system on TLS interface have their source MAC addresses learned and associated with the interface so that frames from the network that come in on other TLS bridges in the VLAN can be sent to the correct interface as shown in [Figure8](#).

Figure 8: With TLS bridges all interfaces learn on ingress

Configure a TLS bridge

This example adds VLAN members to the VLAN 100 to create a community of traffic on the same VLAN.

Configuring a TLS bridge

- 1 For each connection to the TLS bridge add, a *tls* bridge interface

```
zSH> bridge add 1-3-7-0/eth tls vlan 100
```

TLS bridges can be thought of as a community since they share traffic much in the way a physical LAN shares traffic.

- 2 For each connection to the TLS bridge, add a *tls* bridge interface:

```
zSH> bridge add 1-6-5-0/eth tls vlan 100
zSH> bridge add 1-3-16-0/eth tls vlan 100
zSH> bridge add 1-4-17-0/eth tls vlan 100
```

The TLS bridge interfaces with VLAN 100 will all work together as one TLS bridge.



Note: When you do not specify *untagged*, *tagged*, or *staged* to the bridge interface, the interface will use the default for TLS bridges, which is *untagged*.

A note about bridge add parameters

As the example above shows, the **bridge add** command designates the bridge interface type and the VLAN. The **bridge add** command also defines which card and port to add the bridge interface.

The **bridge add** command defines which card, port and interface in the SLMS device by the *shelf-slot-port-subport (or interface)/transport type* syntax. *shelf* is always 1.

For the MALC and the MXK, *slot* is the physical slot where the card resides. For fixed units, like the MALC XP, Raptor XP and EtherXtend the slot is 1. *Port* is the physical port. *Subport* may be different depending on the *transport type*.

For GPON cards, the *transport type* is *gpon* and the subport is the GEM port. For Active Ethernet cards, the transport type is *eth* as in the example above and the subport is the logical interface. You may have multiple logical interfaces per port and the *subport* parameter identifies the logical interface.

Asymmetric bridges

Asymmetric bridges are made up of one uplink and at least one downlink or intralink.

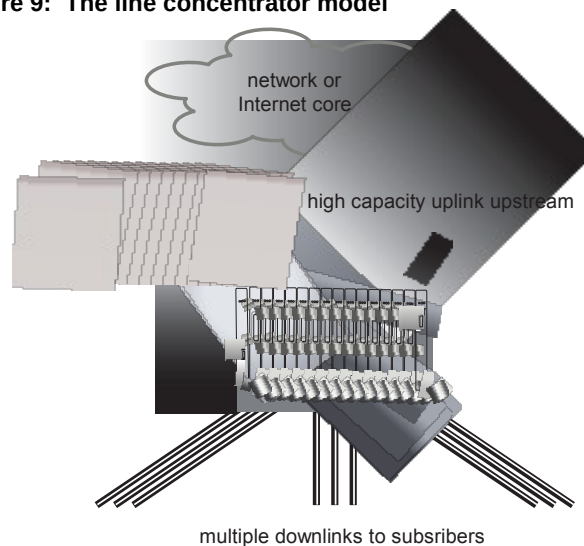
A single asymmetric bridge may use all three asymmetric bridge interface types — uplink, downlink, and intralink — however, a single bridge may only have one uplink. MXKs may have multiple intralinks per bridge, but other SLMS devices may only have one intralink. There may be multiple downlinks.

Most commonly there is one uplink and multiple downlinks as you would have with a line concentrator which splits a high capacity upstream link into multiple lower capacity downstream links.

Intralink bridge interfaces are used for subtending other devices, usually other MXKs or MALCs. Intralinks have different learning behavior than uplinks or downlinks.

When setting up Internet access for multiple subscribers you configure the MXK as a line concentrator. With the line concentrator model you create an asymmetric bridge with a high capacity link upstream configured to be the uplink, and have many downlinks configured for the subscribers.

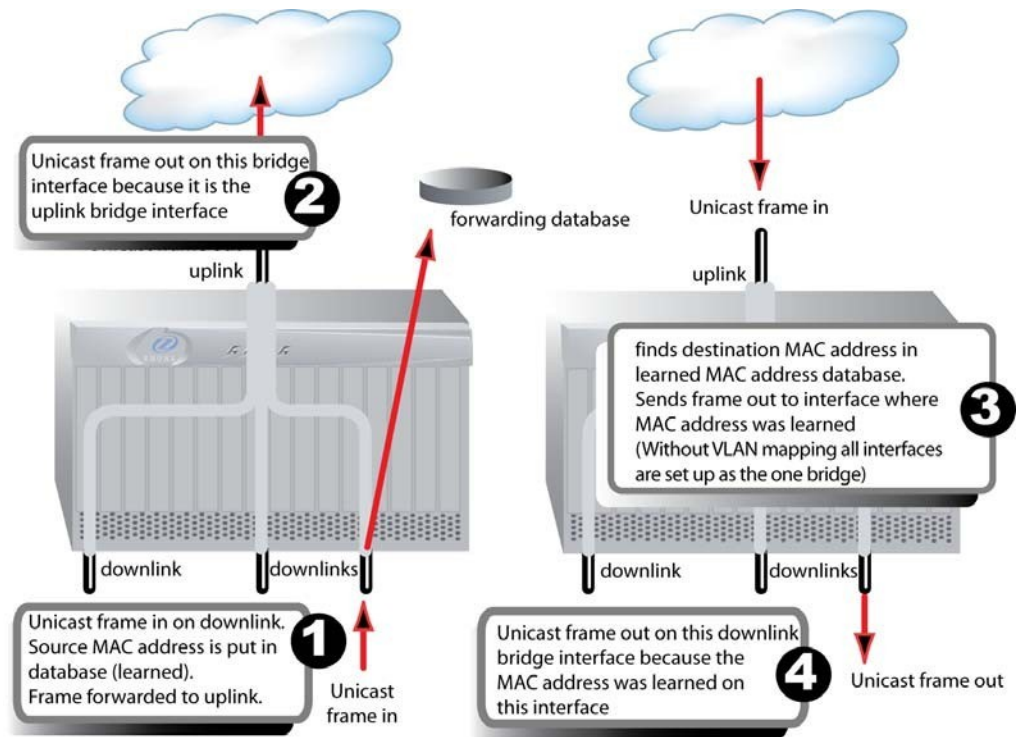
Figure 9: The line concentrator model



When a frame is received on a downlink bridge interface the source MAC address is learned and is put in the forwarding table along with the bridge interface and the VLAN on which the frame was received on. All frames

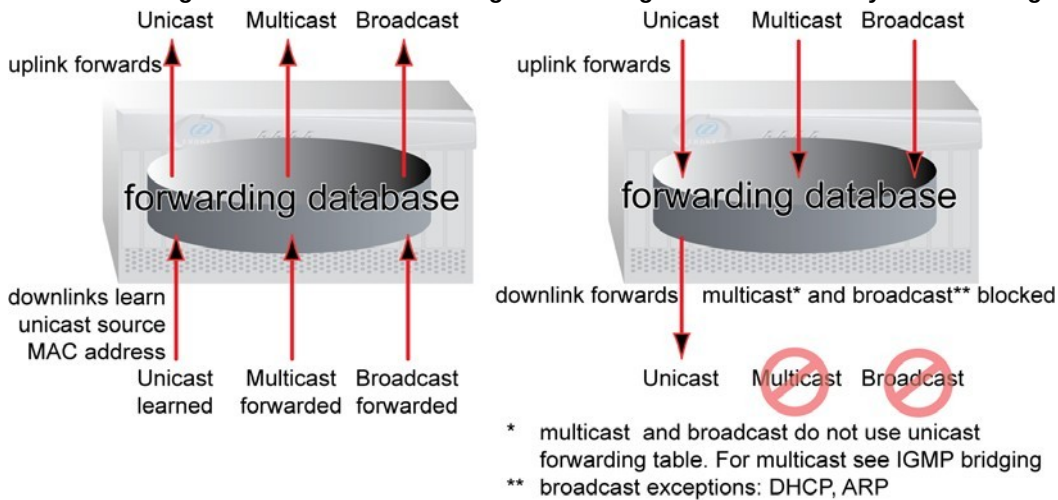
whether unicast, multicast or broadcast received on downlinks are forwarded to the uplink as shown in [Figure10](#).

Figure 10: Unicast forwarding and learning behavior for uplinks and downlinks



Unlike frames received on a downlink interface, when a unicast frame is received on an intralink bridge interface there is no learning and the frame is forced out the uplink as shown in [Figure11](#).

Figure 11: Unicast forwarding and learning behavior for an asymmetric bridge



When frames ingress on an uplink the behavior of an asymmetric bridge is more complex.

When a unicast frame (a frame that is supposed to go to one address) is received on the uplink bridge interface and the address matches a learned MAC address, then the frame is forwarded to that address. Unknown unicast frames received on the uplink are discarded. (Unless there is an intralink, then unknown unicasts are sent on the intralink).

Broadcast frames have a special code in the address portion of the frame which identify it as a broadcast frame. These frames are normally duplicated and sent to all devices.

DHCP servers provide a pool of IP addresses, and upon request provide an IP address for a device. When a MXK acting as a DHCP relay agent receives a broadcast DHCP OFFER message on the uplink from a remote DHCP server the broadcast messages are forwarded to the MAC address if customDHCP is set to true. Otherwise, the broadcast DHCP messages are filtered.

Multicast is used when the same data is required by a group of clients at the same time. Unlike broadcast which sends to all devices, multicast provides content to a limited number of devices simultaneously. A common use of multicast would be video services. Receiving, duplicating and transmitting frames for high quality video to a large number of devices is processing time and capacity intensive. In multicast the number of recipients is guided by the multicast clients requesting to receive the multicast.

In an asymmetric bridge the general rule is that the source address of frames received on the downlinks are learned and the frames are sent out the uplink. Unicast frames received on the uplink are forwarded if found in the forwarding table, discarded if not. Multicasts and broadcasts received on the uplink are not forwarded with the DHCP and ARP exceptions noted above.

Custom ARP

Broadcast frames received on the uplink bridge interface in an asymmetric bridge are blocked. Address Resolution Protocol (ARP) and Dynamic Host Configuration Protocol (DHCP) both are broadcast frames that use the special broadcast code in the address portion of the Ethernet frame but they are dealt with as exceptions.

ARP looks up an IP address in a database which maintains learned IP addresses. In this way ARP is actually a mixture of level 2 (Logical Link with MAC addresses) and Level 3 (Network IP with IP addresses). If the frame is an ARP frame, then the SLMS device compares and filters the requested IP address with the current forwarding table.

How ARP frames are handled is dependent on the **customARP** parameter in the **bridge-interface-profile** which is normally set by default and not needed to be altered.

- When the **customARP** parameter is **false**, the ARP packet is sent out the bridge interface regardless of the whether a match is found for the requested IP address.

- When the **customARP** parameter is **true** and there is a match, the ARP broadcast is forwarded out the interface that has the appropriate host. This host will then reply to the ARP with a standard response.
- When the **customARP** parameter is **true** and there is a match, then the ARP is filtered and the MXK will flood the broadcast out all other bridge interfaces



Note: The MXK has different behavior from all other SLMS devices in respect to how MXK responds to the unmatched ARP message on asymmetric bridges. Rather than flood the unmatched ARP message out on all bridge interfaces, other SLMS devices will drop the unmatched ARP frame as if it were a non-ARP broadcast.

By default **customARP** is set to **true** for Uplinks and false for downlinks and intralinks.

The **customARP** parameter is also set to false for TLS bridge interfaces. For TLS bridges on all SLMS device broadcast packets are broadcast; there is no broadcast filtering.

Configure an uplink bridge

For an uplink bridge you specify an uplink, a bridge-path to send frames received on the downlinks. You must also designate a VLAN ID to match each VLAN ID configured on the downlinks.

Configuring an uplink bridge

- 1 Add a bridge interface on the uplink card.

```
zSH> bridge add 1-a-2-0/eth uplink vlan 500 tagged
Adding bridge on 1-a-2-0/eth
Created bridge-interface-record ethernet2-500/bridge
```

The “1-a-2-0” string defines the shelf-slot-port-interface. This **bridge add** command defines the interface on port 2 of the card in slot *a*, as an Ethernet uplink bridge.

- 2 Add a default bridge path so that all of the traffic on the downlinks with VLAN 500 can be forwarded out of the uplink.

```
zSH> bridge-path add ethernet2-500/bridge vlan 500 default
Bridge-path added successfully
```

The **bridge-path add** command defines where to send traffic from the downlinks. The default option means that all traffic on VLAN 500 from the downlinks are forwarded out the uplink with VLAN 500 designated as the default. The display from the **bridge add** command (or the **bridge show** or **bridge showall** commands) describe the identifier of the bridge interface. Notice that the VLAN is displayed.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
dwn	Tagged 400	ipobridge-400/bridge	UP	D	00:01:47:1a:fe:64
upl	Tagged 500	ethernet2-500/bridge	DWN	S	VLAN 500 default



Note: The MXK does not support the **global** keyword. For each VLAN or SLAN, you must define a default bridge uplink using the **default** keyword. For each VLAN or SLAN you must define the bridge-path as an intralink using the **intralink** keyword.

3 Add downlink bridge interfaces.

```
zSH> bridge add 1-13-1-0/eth downlink vlan 500
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth/bridge
```

```
zSH> bridge add 1-13-2-0/eth downlink vlan 500
Adding bridge on 1-13-2-0/eth
Created bridge-interface-record 1-13-2-0-eth/bridge
```

Intralinked SLMS devices

Intralinks basically daisy chain SLMS devices. The intralink basically takes all frames that cannot be forwarded to a destination.

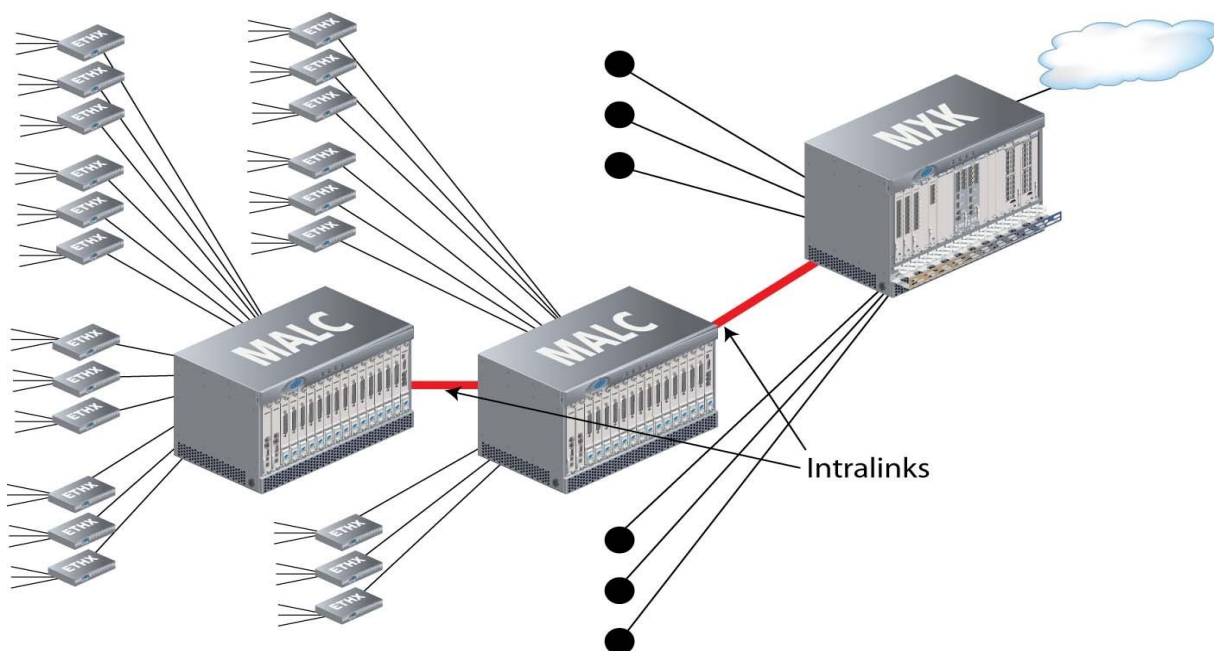
The common case for an asymmetric bridge has the downlinks learning on sending and the uplinks forwarding on reception from outside of the MXK. If a frame is received on a downlink, the MAC address is learned. If the frame in on the uplink has a known address it is forwarded to the downlink that has that address. If the frame is unknown it is discarded.

In a case where you have multiple line concentrators linked, one below another, it is possible for the forwarding table on the head MXK in the chain or the upper MXKs to grow to an unmanageable size because they would be learning the MAC addresses of all devices downstream.

Intralink bridge interfaces, rather than learning the addresses connected to the intralink interface as they would from a downlink, merely send all frames from the intralink interface to the uplink without learning. The reciprocal behavior is that frames with unknown addresses received on the uplink interface are sent down the intralink interface.

[Figure12](#) shows downlinks to EtherXtends and intralinks from an MXK to subtended MALCs. The intralink provides the means to forward all unknown frames received on the uplink to the intralink; the head device for the intralink does not need to learn the frames received on the intralink.

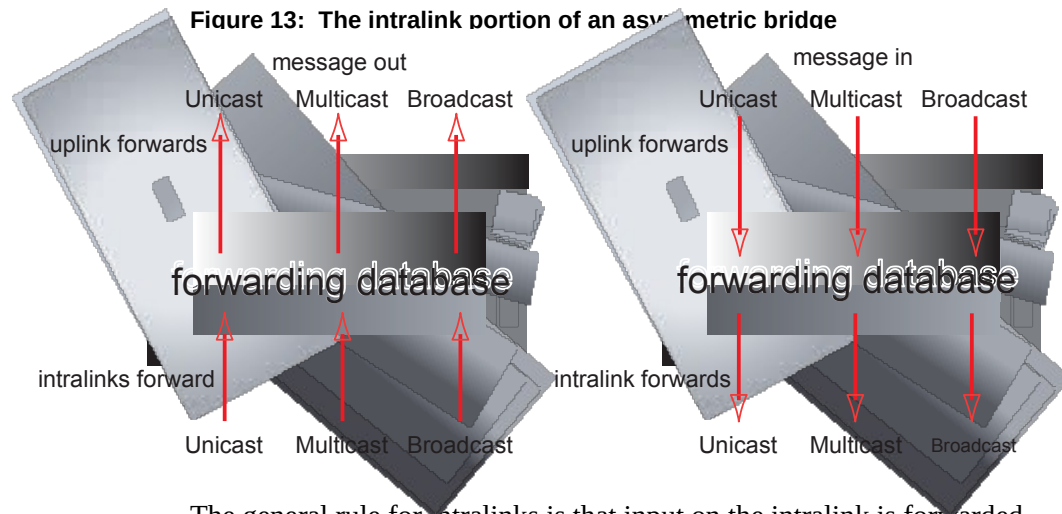
Figure 12: Line concentrator model with intralinks



An intralink bridge interface is used in conjunction with an uplink bridge interface, where the uplink bridge is the path upstream to the network. The intralink interface forwards traffic with unknown MAC addresses or multicasts to the uplink bridge-path without attempting to learn the addresses of the attached devices or network. Traffic coming into the intralink interface is forwarded to the uplink regardless of the destination MAC address. Broadcasts, multicasts, and unicasts (known and unknown) will be sent out the default interface, which is the uplink bridge for the VLAN.

In other words source addresses from an intralink interface are not learned, so the database of learned addresses will not add the address. Likewise when an unknown unicast frame is received on the uplink interface it will be transmitted to the intralink interface. Somewhere down the chain, the address may be known. Intralinks are normally used in conjunction with uplinks and can be used with downlinks.

Like uplinks, intralink bridge interfaces require the additional configuration of a bridge path. The bridge path sets a default intralink path for the specific VLAN onto the intralink bridge. If an intralink is missing the bridge path, traffic will not flow across the asymmetric VLAN.



The general rule for intralinks is that input on the intralink is forwarded without the source address being learned. All frames with unknown addresses are forwarded to the intralink interface.

Configure intralinked MXKs

This example adds an intralink bridge interface to an asymmetric uplink/downlink bridge. The uplink bridges

Configuring MXK intralinks

Configure the MXK for interlinked bridges.

- 1 Add bridge interfaces on the uplink card. For each VLAN ID designated on a downlink, there must be an uplink with the corresponding VLAN ID.

```
zSH> bridge add 1-a-2-0/eth uplink vlan 100
Adding bridge on 1-a-2-0/eth
Created bridge-interface-record ethernet2-100/bridge
```

```
zSH> bridge add 1-a-2-0/eth uplink vlan 200
Adding bridge on 1-a-2-0/eth
Created bridge-interface-record ethernet2-200/bridge
```

- 2 Add the bridge path for the uplink with VLAN 100 and the keyword *default*.

```
zSH> bridge-path add ethernet2-100/bridge vlan 100 default
Bridge-path added successfully
```

```
zSH> bridge-path add ethernet2-200/bridge vlan 200 default
Bridge-path added successfully
```

The *default* keyword designates that all unknown unicast frames will be sent to the intralink rather than discarded as with an asymmetric bridge without an intralink.



Note: The MXK does not support the keyword *global*. For each VLAN or SLAN, you must define a default bridge uplink using the keyword *default*.

3 Add downlink bridges for devices downstream from the MXK.

```
zSH> bridge add 1-13-1-0/eth downlink vlan 100
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth/bridge
```

```
zSH> bridge add 1-13-2-0/eth downlink vlan 200
Adding bridge on 1-13-2-0/eth
Created bridge-interface-record 1-13-2-0-eth/bridge
```

4 Create a bridge interface for the intralink with a VLAN ID.

The intralink can be between the MXK and a subtended MXK, MALC, or SLMS device. Then add the bridge path for the intralink.

```
zSH> bridge add 1-13-3-0/eth intralink vlan 444
Adding bridge on 1-13-3-0/eth
Created bridge-interface-record 1-13-3-0-eth-444/bridge
```

```
zSH> bridge-path add 1-13-3-0-eth-444/bridge vlan 444 default
Bridge-path added successfully
```

This command mainly defines the behavior that source addresses from the intralink will not be learned.



Note: The MXK does not support the **global-intralink** keyword. For each VLAN or SLAN, you must define the bridge-path as an intralink using the **intralink** keyword.

This command defines the behavior that any frames with unknown addresses will be sent to the interlink with VLAN ID 444.

5 Create the uplink bridge for the intralink with the same VLAN ID for traffic to be passed to the network.

```
zSH> bridge add 1-a-3-0/eth uplink vlan 444
Adding bridge on 1-a-3-0/eth
Created bridge-interface-record ethernet3-444/bridge
```

```
zSH> bridge-path add ethernet3-444/bridge vlan 444 default
Bridge-path added successfully
```

6 Verify the bridges created.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
dwn	Tagged 400	ipobridge-400/bridge	UP	D	00:01:47:1a:fe:64
upl	Tagged 100	ethernet2-100/bridge	DWN	S	VLAN 100 default
upl	Tagged 200	ethernet2-200/bridge	DWN	S	VLAN 200 default

```

dwn          100 1-13-1-0-eth/bridge      DWN
              200 1-13-2-0-eth/bridge      DWN
int Tagged 444 1-13-3-0-eth-444/bridge     DWN S VLAN 444 default
upl Tagged 444 ethernet3-444/bridge        DWN S VLAN 444 default

```

7 Verify the bridge paths.

```
zSH> bridge-path show
```

VLAN/SLAN	Bridge	Address
100	ethernet2-100/bridge	Default
200	ethernet2-200/bridge	Default
444	1-13-3-0-eth-444/bridge	Default
444	ethernet3-444/bridge	Default

Tagging operations

This section describes VLAN and SLAN tagging operations including:

- [Overview, page211](#)
- [Common tagging operation scenarios, page214](#)
- [Tagging options, page221](#)

Overview

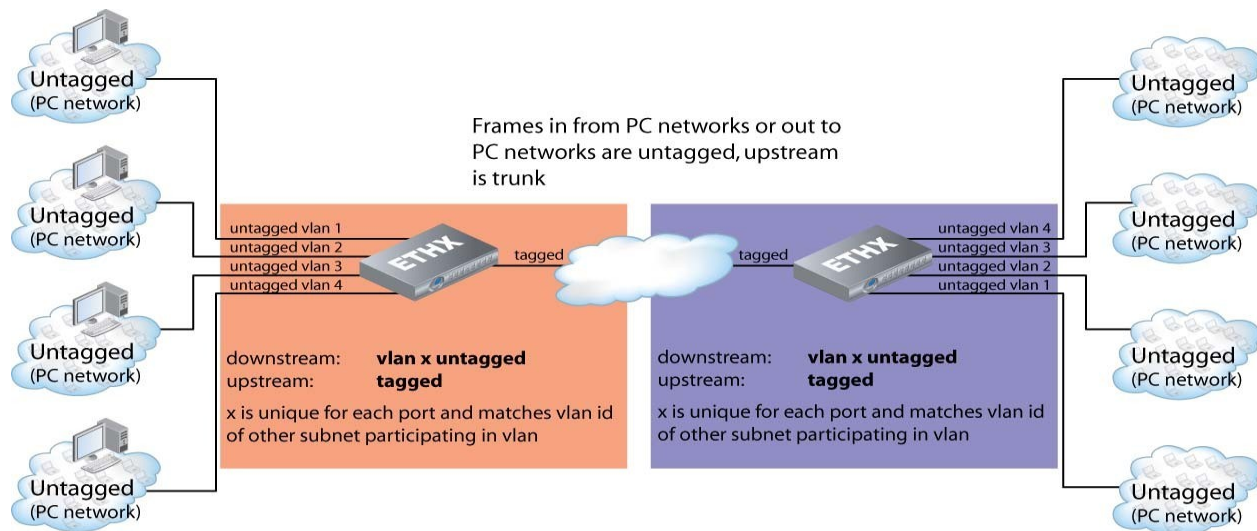
VLANs and SLANs (see [VLANs and SLANs, untagged, tagged and staged, page193](#) for information about VLANs and SLANs) define the bridge to which an incoming frame belongs. The bridge type — as discussed in [Section 5, SLMS bridge types](#) — determines the forwarding behavior for the bridge. In conjunction with the forwarding and learning characteristics from the bridge types, you can also configure tagging operations.

Tagging operations provide the ability to configure interfaces for ingress filtering, VLAN/SLAN promotion, egress, and/or stripping.

Usually these tagging operations — ingress filtering, promotion, egress and/or stripping — are configured on downstream interfaces. Defining whether a bridge interface should be untagged, tagged or staged depends on what the devices connected to the interface are expecting.

Zhone uses an extremely flexible mechanism for configuring tagging operations. Before discussing the various combinations which are possible, it is important to understand common cases, including the most common case — VLAN tagging for PC networks.

Figure 14: VLAN tags can be used to organize subnets



You can add a VLAN tag to all frames coming in from a PC network which has untagged Ethernet frames. However you want the PC network to be part of a virtual LAN with another remote PC network, so you configure the downstream bridge interface to accept the untagged frames and add a tag. Zhone uses the term *promotion* to signify adding the tag. The frames are then tagged frames and are sent out the upstream bridge interface tagged and directed to the remote PC network. The upstream bridge is a trunk line.

Likewise on receiving a frame from the remote PC network (which has the same VLAN tag), the frame is received on the uplink and forwarded to the proper downstream link because the VLAN ID matches (and assuming the destination MAC address of the unicast frame matches a learned MAC address). However the PC network does not accept tags, so the VLAN tag is removed and the frame is forwarded to the device with the proper MAC address. Zhone uses the term *stripping* to signify removing VLAN and/or SLAN IDs.

In [Figure14](#), The EtherXtend 34xx family (An SLMS based CPE/DSLAM, so the commands are the same as the MALC or MXK), are providing VLAN tags so on the other side of the cloud the frames may be forwarded to the proper VLANs as defined by the other EtherXtend. In [Figure14](#), the cloud may just be the cabling between two EtherXtends connected back to back; the cloud could also be a whole network of subtending MALCs, MXKs, the Internet, but the basic VLAN tagging is being done at the EtherXtend devices at the network edge. The 34xx family has Ethernet ports to subscribers and EFM-SHDSL ports upstream.

In the example from [Figure14](#), the upstream interfaces are tagged with no VLAN ID designated. The downstream interfaces are untagged and given a VLAN ID which identifies which port (and hence which PC network) the frames received on these interfaces came from. This VLAN definition describes which VLAN tag to insert on ingress, and that VLAN ID upon receiving on the upstream interface on the remote EtherXtend defines which

downstream port to forward the frame. Since the downstream interface is untagged, the VLAN ID tag is stripped off and the frame sent out to the remote PC network.



Note: This example does not describe whether the bridges are asymmetric bridges or TLS bridges.

The four VLAN operations work together and are implied in the **bridge add (bridge modify)** command.

- Ingress filtering is the ability to have the bridge interface accept only frames with certain types of VLAN/SLAN tags.
- VLAN/SLAN promotion is the ability to add tags to a Ethernet frame. As with the example in [Figure14](#), the VLAN tag defines membership in a VLAN (VLAN/SLAN defines membership with two tags).
- Egress is the reciprocal of ingress filtering and designates where to forward the frame based on VLAN, SLAN, or VLAN/SLAN tags. If a frame is received into the device and possibly promoted, then needs to find the other bridge interface(s) for egress.
- Stripping is the reverse of promotion. Stripping is removing the VLAN, SLAN or VLAN/SLAN tags.

Promotion and stripping always occur together. Filtering on ingress assumes the incoming frames already have at least one tag; you may filter on VLAN and also promote an SLAN. Receiving the internally forwarded frame to the egress assumes that the frame either has been received with tags or has been promoted to have tags.

The discussion of VLANs and tagging uses a tabular format for describing the above operations, [Common tagging operation scenarios on page214](#) and [Tagging options on page221](#), using graphic representations to show the changes in frames as they are received on an interface forwarded to an egress interface and possibly promoted or stripped.

Table 2: Possible tags bridge interfaces may be configured to accept

Possible Tags	Graphic representation
untagged	
tagged with unknown VLAN ID	
tagged with known VLAN ID	
tagged with unknown VLAN ID and SLAN ID	
tagged with unknown VLAN ID and known SLAN ID	
tagged with known VLAN ID and SLAN ID	

Zhone does not support tagged with known VLAN ID and unknown SLAN ID.



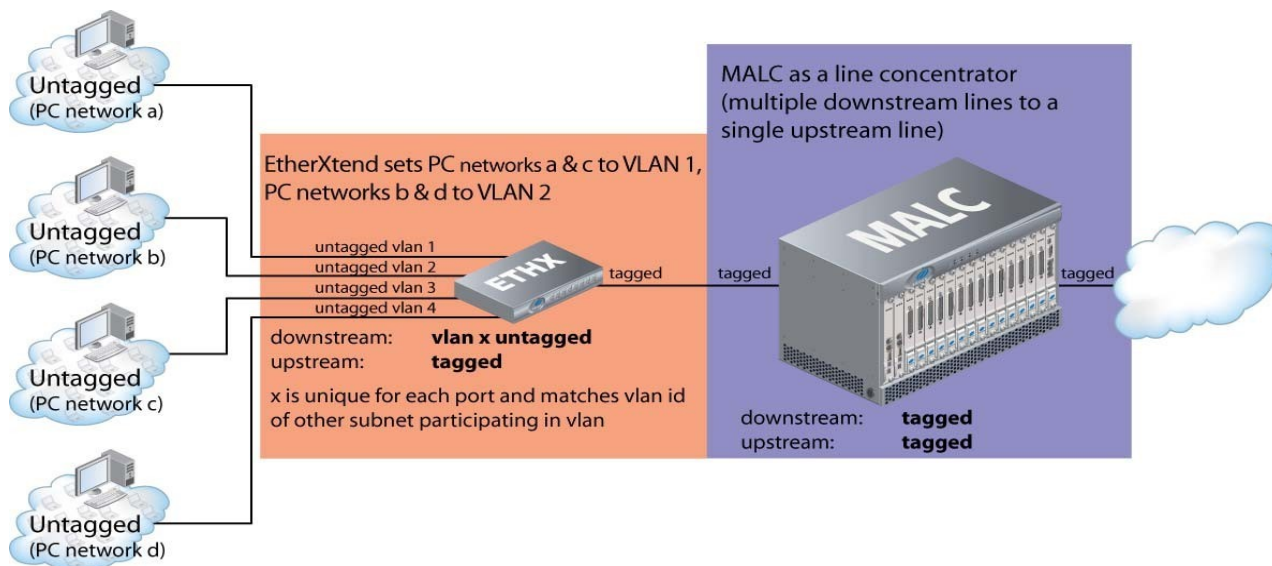
Note: The MXK does not support tagged frames with unknown VLAN and unknown SLAN.

The frames which come into the MXK are untagged, tagged and double tagged.

Common tagging operation scenarios

All SLMS devices support promoting tags. How you define the next level upstream from the edge of the network depends on your network architecture. In [Figure15](#), the MALC is the next level up from the EtherXtend and acts as line concentrator. Imagine there is an MXK upstream from the MALC as in [Figure18](#). The example shows only VLAN tagging, but any of the SLMS devices could promote an s-tag, depending on what is necessary in the application or the overall network architecture.

Figure 15: EtherXtend providing edge tagging, MALC as line concentrator



[Figure15](#) shows promoting untagged frames on the downstream interface (and so filtering to that interface when a frame with that VLAN id is received on the upstream interface — given that the other bridging fundamentals are met, such as the MAC address as well as the VLAN id match in the forwarding table if it is a downlink).

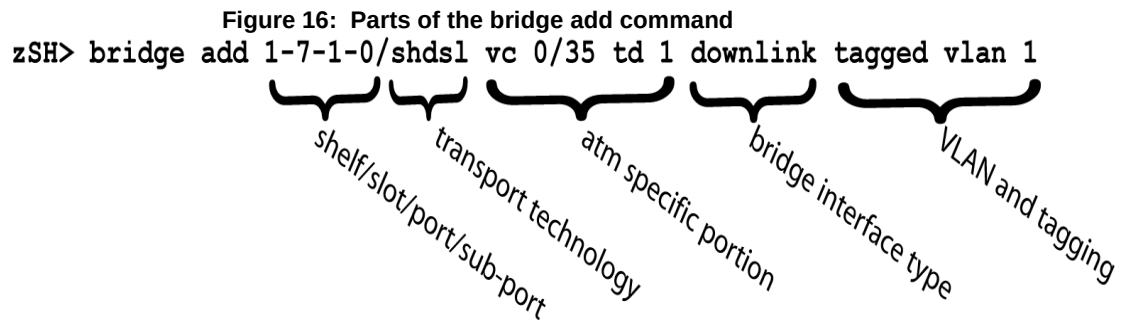
The untagged frame is accepted on the downstream interface, then it is promoted by inserting a VLAN ID. The upstream is tagged, so the tagged frame is sent out the upstream interface.

In order to complete the overlay with tagging and bridge types it helps to understand the following: the tagged frame will go out the uplink if part of an

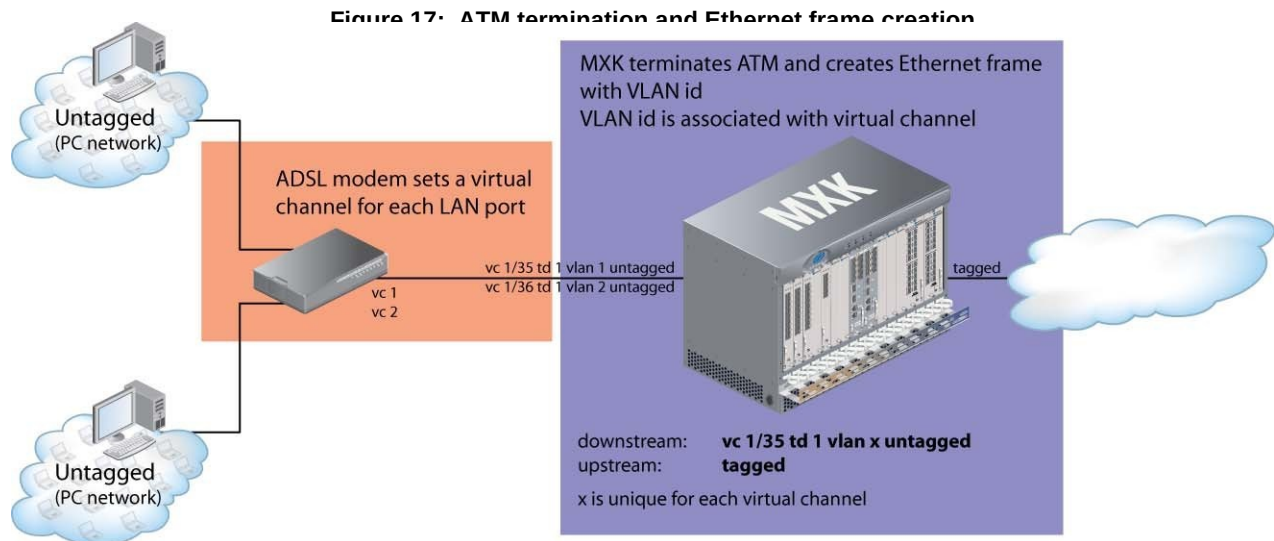
asymmetric bridge; if a TLS bridge the frame will go where the forward table says it should go — the upstream interface if the MAC address matches. If the MAC address does not match addresses in the forwarding table the frame (an unknown unicast) would go out the upstream interface (along with the other participating bridge interfaces except the ingress bridge interface) since with TLS unknown unicasts are flooded out all member interfaces of the bridge

A good way to learn tagging fundamentals is by exploring some of the common scenarios. [Figure14](#) shows promoting (and stripping) VLAN tags at the network edge. [Figure15](#) shows that same promotion at the edge, but now a line concentrator (in the example a MALC) distributes access from many downstream lines to a trunk. These multiple downstream subscriber lines could be from different transport technologies. In [Figure15](#) the EtherXtend uses EFM SHDSL where the frames are already Ethernet frames. For the next example, [Figure17](#), the downstream devices could also be ADSL based (along with EFM SHDSL or other transport media at the same time).

ADSL technologies are based on ATM virtual connections. Another example of VLANs is terminating ATM from an xDSL modem and creating an Ethernet frame. In this case, the VLAN id is associated with the virtual channel. The ATM virtual connections can then be terminated and the data put into Ethernet frames with VLAN tags corresponding to the ATM virtual channel.



ADSL termination/Ethernet frame creation is a good example to show the parts of the **bridge add** command. Portions of the command define the bridging characteristics discussed in this chapter. The command also includes the transport technology and any associated information, such as the ATM specific portion for xDSL transport media.



Look at edge tagging in a tabular format to see that this same basic promotion concept works for different network.

The frame received on the downstream interface is untagged. Reading left to right, that frame is promoted to have a VLAN ID depending on the interface where the frame was received. The upstream interface is tagged, so a frame with a VLAN ID (but not double tagged) is forwarded to that interface. Since the bridge interface is tagged there is no stripping.

Downstream interface definition	Ingress frame	Promotion?	Egress frame	Egress stripping?	Upstream interface definition	Description
untagged VLAN x					tagged	Very common case....(1) ATM (xDSL termination) xDSL LC with vc5 (voice to VLAN 5), vc6 (data to VLAN 6) which can be separated by a VLAN switch upstream from the malc to a softswitch (VLAN 5), BRAS (VLAN 6). (2) tagging downstream separate networks based on downstream port.

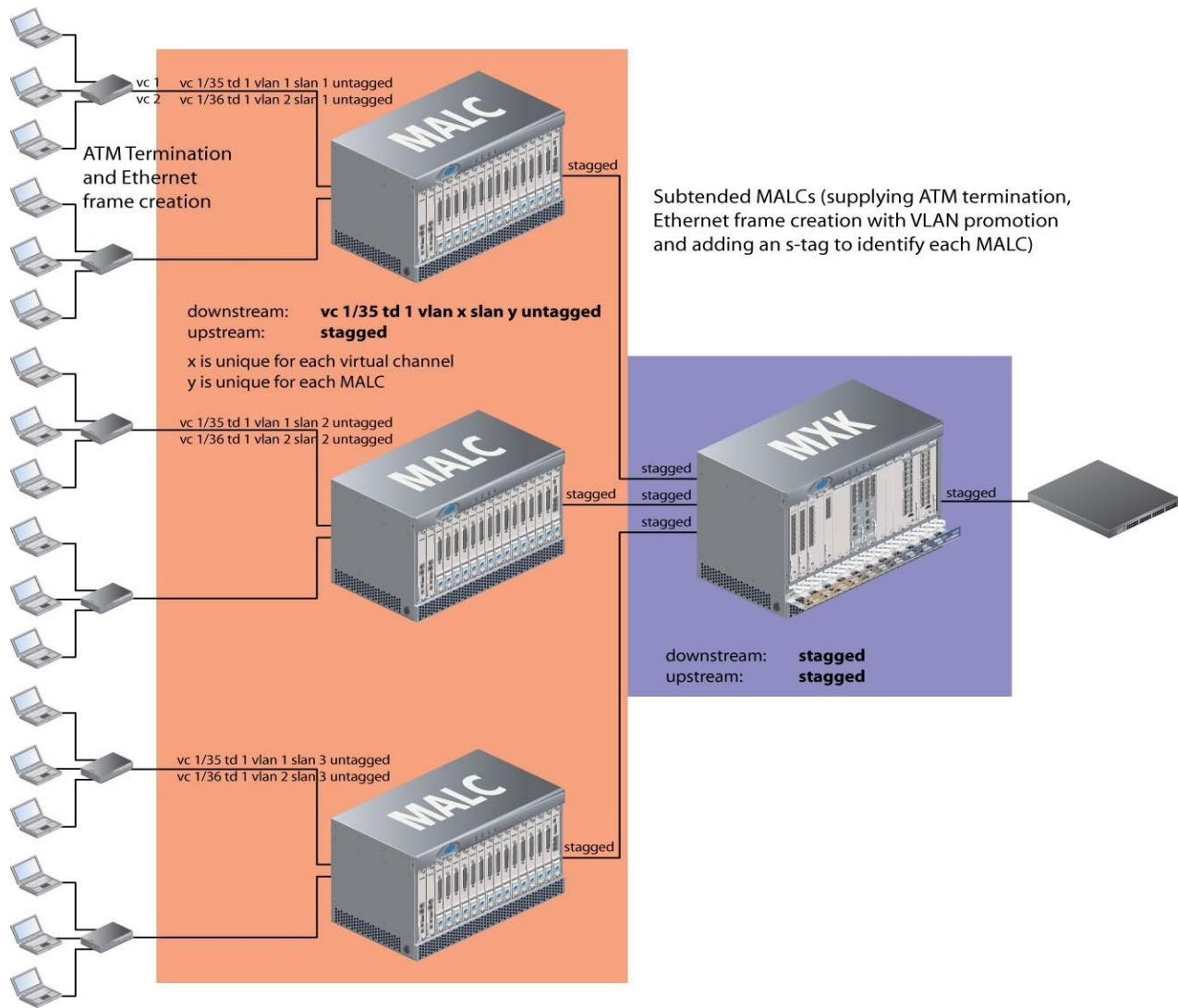
A frame on the upstream interface makes a reciprocal trip. A tagged frame is accepted on the upstream interface. Since no VLAN is defined it accepts all single tagged frames (so any VLAN ID). There is no promotion. The frame is forwarded to the bridge interface with the VLAN ID which matches the VLAN ID of the Ethernet frame. The egress interface is also untagged, so the VLAN ID is stripped out and the frame is sent to the network.

Upstream interface definition	Ingress frame	Promotion?	Egress frame	Egress stripping?	Downstream interface definition	Description
tagged					untagged VLAN x	Tagged frame on uplink is forwarded based on VLAN ID that defines which interface, then VLAN stripped for untagged network.

In this case multiple interfaces with the same VLAN are not being discussed, though that is a very common scenario. For the sake of discussion here, MAC addresses are found in the forwarding table for the egress interface.

[Figure 18](#) describes the next step upstream and describes double tags (the second tag are also called s-tags). In a subtended scenario you can add an s-tag for tracking the origination of the frame, perhaps by department. The example in [Figure 18](#) shows the double promotion of tags. The example shows the MALC providing ATM termination and the linkage to a VLAN ID and the promotion of an s-tag as well.

Figure 18: Q in Q supports adding a second tag

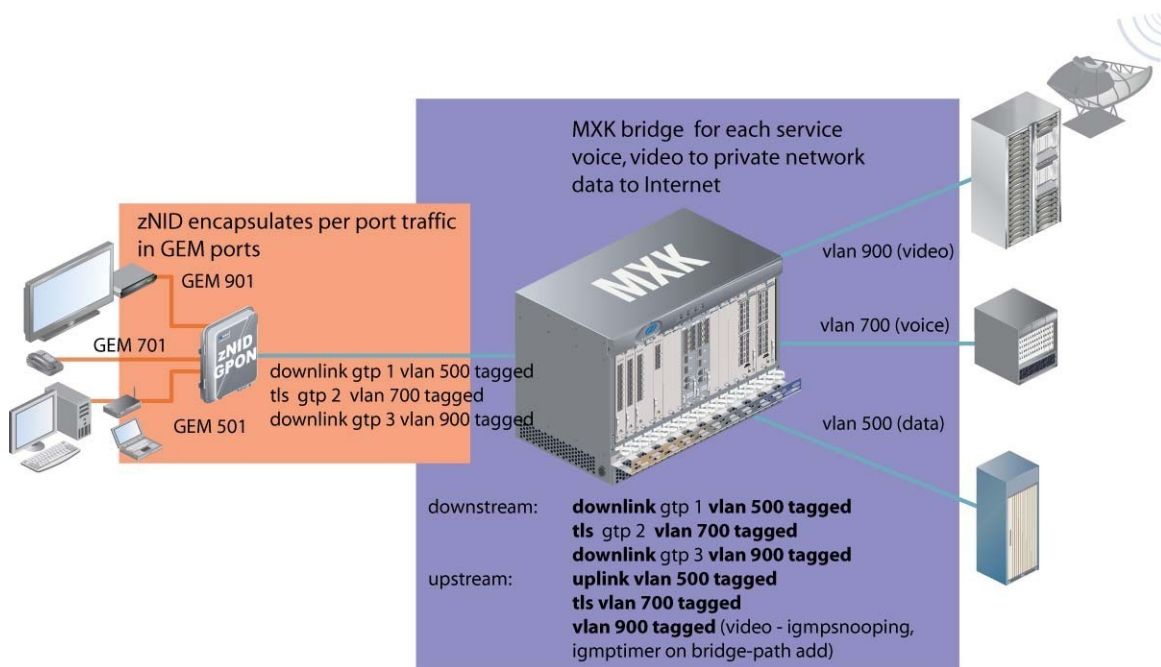


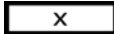
In the table describing the subtended MALCs, ingress frames received on the downstream bridge interface have both VLAN and SLAN IDs promoted. In this case the VLAN ID defines the ATM virtual channel. The SLAN ID designates from which MALC the frame originated.

Downstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Upstream interface definition	Description
untagged VLAN x SLAN y					tagged	The MALC provides ATM termination for xDSL modem and provides a second tag for identify the source of the frame.

Upstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Downstream interface definition	Description
staged					untagged VLAN x SLAN y	Frame comes in staged and is filtered to proper downstream link

The uplinks can be separated by VLAN which is a common scenario (see [VLANs and SLANs, untagged, tagged and staged](#)). Normally in a triple play scenario you would have separate VLANs for video or voice services. In this way you can keep known traffic separate for defining QoS prioritization or other bridge additions as provided by packet rules.

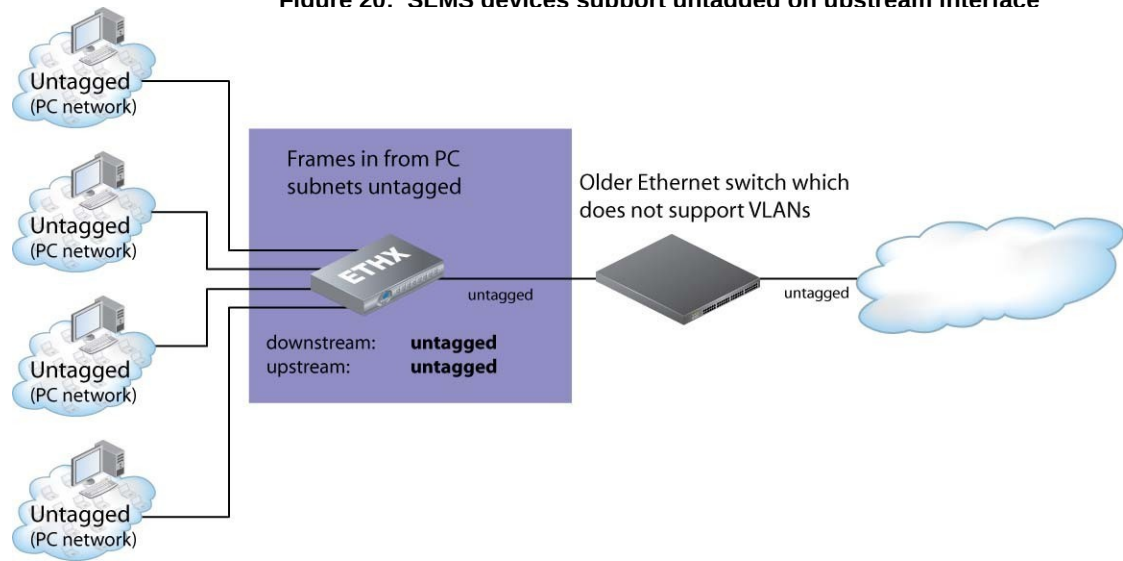


Downstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Upstream interface definition	Description
tagged VLAN x					tagged VLAN x	This combination, tagged VLAN x - tagged VLAN x, gives full control for bridge rules on both ingress and egress

Upstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Downstream interface definition	Description
tagged VLAN x	x	x	x	x	tagged VLAN x	

The flexibility of the SLMS tagging mechanism works for many scenarios. Not only do the MXK and MALC support many transport media technologies, but they can also support every level of tagging, both on the downstream interface as well as on the upstream interface.

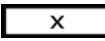
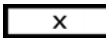
Figure 20: SLMS devices support untagged on upstream interface

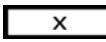
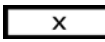


Downstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Upstream interface definition	Description
untagged					untagged	This competition, untagged - untagged provides a pass through mechanism when there is an Ethernet switch upstream which does not support VLAN tagging.

Upstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Downstream interface definition	Description
untagged					untagged	

To separate untagged information where you have other traffic which you would have as VLAN 0 (untagged frames which do not belong to a VLAN), you could tag on ingress and strip that tag on egress.

Downstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Upstream interface definition	Description
untagged VLAN x					untagged VLAN x	This combination, untagged VLAN x - untagged VLAN x also provides a pass through mechanism when there is an Ethernet switch upstream which does not support VLAN tagging, however this combination also provides the means for directing multiple untagged VLAN traffic through the device.

Upstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Downstream interface definition	Description
untagged VLAN x					untagged VLAN x	

In this example there is promotion, filtering and stripping to provide an extra option.

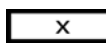
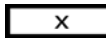
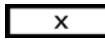
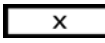
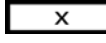
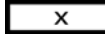
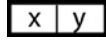
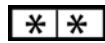
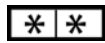
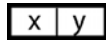
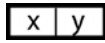
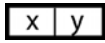
Tagging options

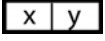
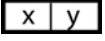
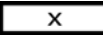
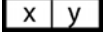
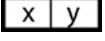
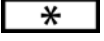
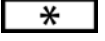
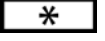
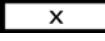
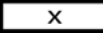
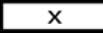
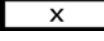
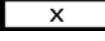
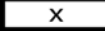
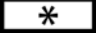
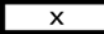
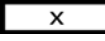
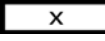
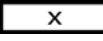
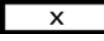
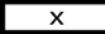
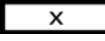
The following table shows all the possible combinations available with the **bridge add/bridge modify** command. It should be understood that there are more possible configurations provided by the mechanism than are required in field deployment, however the flexibility of the mechanism allows for instant configuration of a wide range of requirements. Great care should be taken

when defining tagging operations. Please consult Zhone Global Service & Support or your sales representative for assistance.

Read the table from left to right, row by row. The first column shows the VLAN portion of the bridge add command. Ingress frame shows which type of tagging the bridge interface accepts — untagged, tagged x (tagged with a specific VLAN ID), tagged (tagged without a specified VLAN ID; any VLAN, shown by the wildcard “*”), double tagged with specified VLAN and SLAN (VLAN x SLAN y), double tagged with unspecified VLAN and SLAN (tagged, no VLAN or SLAN specified, shown as “*|*”), and tagged with a known SLAN ID but unknown VLAN ID (tagged SLAN y, shown as “*|y”).

Zhone does not support tagged with known VLAN ID and unknown SLAN ID.

Downstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Upstream interface definition	Possible uses
untagged					untagged	(1) Ethernet switch without VLAN support (2) only 1 service (data only)
untagged VLAN x					tagged	Common case; please see Figure14 , Figure15 , and Figure17
untagged VLAN x					tagged VLAN x	
untagged VLAN x					untagged VLAN x	(1) untagged VLANs on separate subscriber nets to separate networks upstream (2) multiple untagged subscriber nets that you want to make into VLANs downstream from the SLMS device, upstream is Ethernet switch without VLAN support.
untagged VLAN x SLAN y					tagged	Common case, please see Figure18
untagged VLAN x SLAN y					tagged VLAN x SLAN y	Similar to the case above, but can provide unique bridge packets rules based on VLAN/SLAN

Downstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Upstream interface definition	Possible uses
untagged VLAN x SLAN y					tagged VLAN x SLAN y	Not likely to add SLAN ID and strip it off on uplink, unless have separate uplinks for each SLAN. Could add bridge packet rules to specific VLAN/SLAN.
untagged VLAN x SLAN y					untagged VLAN x SLAN y	Not a likely case.
tagged					tagged	Common case... downstream device providing promotion or stripping behavior. Just being a line concentrator
tagged					untagged	Non VLAN switch... an odd case in that taking in all VLANs and stripping on upstream to a non VLAN capable switch is not likely.
tagged					tagged VLAN x	Provide specific egress filtering on uplink.
tagged					untagged VLAN x	Not a likely case.
tagged VLAN x					tagged	ingress filtering on VLAN (packet rules)
tagged VLAN x					tagged VLAN x	Provide specific ingress filtering & egress filtering on uplink based on VLAN.
tagged VLAN x					untagged VLAN x	Non VLAN switch upstream, but filtering only to a specific VLAN for ingress filtering.

Downstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Upstream interface definition	Possible uses
tagged SLAN y	*	* y	* *	* *	staggered	Adding SLAN ID to VLAN ID for tracking, n downstream to 1 upstream.
tagged SLAN y	*	* y	* y	* y	staggered SLAN y	Adding SLAN ID to VLAN ID for tracking, but adding egress packet rules to the specific SLAN.
tagged SLAN y	*	* y	* y	*	tagged SLAN y	Not a likely case.
tagged SLAN y	*	* y	x y	x y	staggered VLAN x SLAN y	Adding packet rules for behavior for specific VLAN.
tagged SLAN y	*	* y	x y	x	tagged VLAN x SLAN y	not a likely case
tagged SLAN y	*	* y	x y		untagged VLAN x SLAN y	not a likely case
tagged VLAN x SLAN y	x	* *	* *	* *	staggered	filtering a specific VLAN on downstream ingress, then adding an SLAN tag for tracking.
tagged VLAN x SLAN y	x	x y	x y	x y	staggered VLAN x SLAN y	not a likely case
tagged VLAN x SLAN y	x	x y	x y		tagged VLAN x SLAN y	not a likely case
tagged VLAN x SLAN y	x	* y	x y	* y	untagged VLAN x SLAN y	not a likely case
tagged VLAN x SLAN y	x	* y	x y	* y	tagged SLAN y	not a likely case
tagged VLAN x SLAN y	x	* y	x y	*	tagged SLAN y	not a likely case
staggered	* *	* *	* *	* *	staggered	subtending with trusted network, edge access device providing tagging and filtering. This device is configured as a line concentrator.

Downstream interface definition	Ingress frame	Promotion?	Egress?	Egress stripping?	Upstream interface definition	Possible uses
staged					staged vlan x SLAN y	subtending with trusted network, but adding individual VLAN/SLAN filtering rules on egress
staged					tagged vlan x SLAN y	not a likely case
staged					untagged vlan x SLAN y	not a likely case
staged					staged SLAN y	not a likely case
staged					tagged SLAN y	not a likely case
staged vlan x	NOT SUPPORTED					
staged SLAN y					staged	not a likely case
staged SLAN y					staged SLAN y	not a likely case
staged SLAN y					tagged SLAN y	not a likely case
staged SLAN y					staged vlan x SLAN y	not a likely case
staged SLAN y					tagged vlan x SLAN y	not a likely case
staged SLAN y					untagged vlan x SLAN y	not a likely case
staged vlan x SLAN y					staged	not a likely case
staged vlan x SLAN y					staged VLAN x SLAN y	not a likely case
staged VLAN x SLAN y					tagged VLAN x SLAN y	not a likely case
staged VLAN x SLAN y					untagged VLAN x SLAN y	not a likely case

Common bridge commands

This section describes:

- [bridge add command, page226](#)
- [Verifying bridge interface settings, page226](#)

bridge add command

The **bridge add** command combines fundamental bridging types, the physical interface with its transportation media specifics, tagging operations and other bridge rule additions such as packet rule records. (See [SLMS bridge types on page198](#), [Physical interface on page192](#), [Tagging operations on page211](#) and [Advanced bridging topics on page260](#) for more detail).

This section describes the generic bridge commands, not the transportation media specifics, nor the advanced topics, but concentrates on the most common uses, not all the available options. Please see [Advanced bridging topics on page260](#) for options or the SLMS CLI reference guide.



Note: Entering general CLI commands on systems with large GPON configurations can take a long time to process. You must be as specific as possible with CLI commands. For example, **bridge flush all** should not be used. Instead, use commands based on the specific interface or MAC address.

Refer to the *MALC CLI Reference Guide* for a detailed explanation of the available bridge commands.

Verifying bridge interface settings

Bridge interfaces are created with the **bridge add** command. View the **bridge-interface-record** profile to investigate issues, or when the bridge-interface-record profile defaults do not provide needed bridging behavior.

To view the current bridge interface settings, enter **get bridge-interface-record interface/type**.

```
zSH> get bridge-interface-record ethernet2-800/bridge
bridge-interface-record  ethernet2-800/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {800}
stripAndInsert: -----> {false}
customARP: -----> {true}
filterBroadcast: -----> {true}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
```

```

forwardToUnicast: -----> {true}
forwardToMulticast: -----> {true}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {true}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

A **bridge-interface-record** profile is a set of parameters. The configuration of the different bridge interface parameters defines the behavior of the bridge interface. Bridge interfaces work together and the combination of the bridge interfaces is considered a bridge.

Refer to the *CLI Reference Guide* for a complete description of the command options and syntax.

Bridge add and bridge-path add

All uplink bridges on the MXK require a VLAN ID. There must be an uplink bridge with a VLAN ID to match any existing downlink bridges with a VLAN ID in order to pass traffic. All uplink bridges default to *tagged* with the **stripAndInsert** parameter set to *false*. This means that the VLAN ID remains and is passed up to the network.

On the MXK, all bridge paths are set to *default*.

Entering the **bridge add** command for uplink and intralink bridges automatically creates the bridge path with the default configuration for that bridge. For configuring additional parameters such as *mcast* and *igmp* for video bridging, enter the **bridge-path add** command with the proper variables.

For example:

```

zSH> bridge-path addethernet4-300/bridge vlan 300 default igmptimer 30 igmpsnooping
enable
    Bridge-path added successfully

```

or

```
zSH> bridge-path add ethernet2-800/bridge vlan 800 default mcast 90 igmptimer  
30
```

Bridge-path added successfully

Settings for asymmetric bridges

[Table 3](#) lists the default asymmetric **bridge-interface-record** settings for the supported bridge options.

Table 3: Default values for asymmetric bridge-interface-record

Parameter	Uplink	Downlink Untagged	Downlink Tagged	Intralink Tagged
vpi	0 for Ethernet interfaces. As specified for other interfaces.	0 for Ethernet interfaces. As specified for other interfaces.	0 for Ethernet interfaces. As specified for other interfaces.	0 for Ethernet interfaces. As specified for other interfaces.
vci	0 for Ethernet interfaces. As specified for other interfaces.	0 for Ethernet interfaces. As specified for other interfaces.	0 for Ethernet interfaces. As specified for other interfaces.	0 for Ethernet interfaces. As specified for other interfaces.
vlanId	As specified	As specified	As specified	As specified
stripAndInsert	False	True	False	False
customARP	True	False	False	False
filterBroadcast	True	False	False	False
learnIP	False	True	True	False
learnUnicast	False	True	True	False
maxUnicast	0	5	5	0
learnMulticast	False	True	True	False
forwardToUnicast	True	False	False	False
forwardToMulticast	True	False	False	False
forwardToDefault	False	True	True	True
bridgeIfCustomDHCP	True	False	False	False
bridgeIfIngressPacketRuleGroupIndex	0000			
valndIdCOS	0	0	0	0
outgoingCOSOption	Disable	Disable	Disable	Disable
outgoingCOSValue	0	0	0	0
s-tagTPID	0x8100	0x8100	0x8100	0x8100
s-tagId	0	0	0	0
s-tagStripAndInsert	True	True	True	True
s-tagOutgoingCOSOption	s-tagdisable	s-tagdisable	s-tagdisable	s-tagdisable

Table 3: Default values for asymmetric bridge-interface-record (Continued)

Parameter	Uplink	Downlink Untagged	Downlink Tagged	Intralink Tagged
s-tagIdCOS	0	0	0	0
s-tagOutgoingCOSValue	0	0	0	0
mcastControlList	As specified	As specified	As specified	As specified
maxVideoStreams	0	0	0	0
isPPPoA	False	False	False	False
floodUnknown	False	False	False	False
floodMulticast	False	False	False	False
bridgeIfEgressPacketRuleGroupIndex:	0000			
bridgeIfTableBasedFilter	NONE(0)	NONE(0)	NONE(0)	NONE(0)
bridgeIfDhcpLearn	NONE(0)	NONE(0)	NONE(0)	NONE(0)

Settings for symmetric bridges

[Table 4](#) lists the default **bridge-interface-record** settings for the supported symmetric bridge options.

Table 4: Default values for TLS bridge-interface-record

Parameter	TLS
vpi	0 for Ethernet interfaces. As specified for other interfaces.
vci	0 for Ethernet interfaces. As specified for other interfaces.
vlanId	As specified
stripAndInsert	True
customARP	False
filterBroadcast	False
learnIP	False
learnUnicast	True
maxUnicast	100
learnMulticast	False
forwardToUnicast	True
forwardToMulticast	False

Table 4: Default values for TLS bridge-interface-record (Continued)

Parameter	TLS
forwardToDefault	False
floodUnknown	True
floodMulticast	True
bridgeIfCustomDHCP	False
bridgeIfConfigGroupIndex	0
valndIdCOS	0
outgoingCOSOption	Disable
outgoingCOSValue	0
s-tagTPID	0x8100
s-tagId	0
s-tagStripAndInsert	False
s-tagOutgoingCOSOption	s-tagdisable
s-tagIdCOS	0
s-tagOutgoingCOSValue	0
mcastControlList:	{}
maxVideoStreams	0
isPPPoA	false
floodUnknown	true
floodMulticast	true
bridgeIfEgressPacketRuleGroupIndex	0
bridgeIfTableBasedFilter	NONE(0)
bridgeIfDhcpLearn	NONE(0)

The **bridge show** command displays the bridge type.

zSH> **bridge show**

Type	VLAN	Bridge	St	Table	Data
upl	Tagged 777	ethernet4-777/bridge	DWN	S	VLAN 777 default
upl	Tagged 888	ethernet5-888/bridge	UP	S	VLAN 888 default
dwn	Tagged 888	1-13-1-0-eth-888/bridge	DWN		
upl	Tagged 999	ethernet8-999/bridge	DWN	S	VLAN 999 default
dwn	400	1-13-1-0-eth/bridge	DWN		
upl	Tagged 400	ethernet4-400/bridge	DWN	S	VLAN 400 default
dwn	Tagged 400	ipobridge-400/bridge	UP	D	00:01:47:1a:fe:64

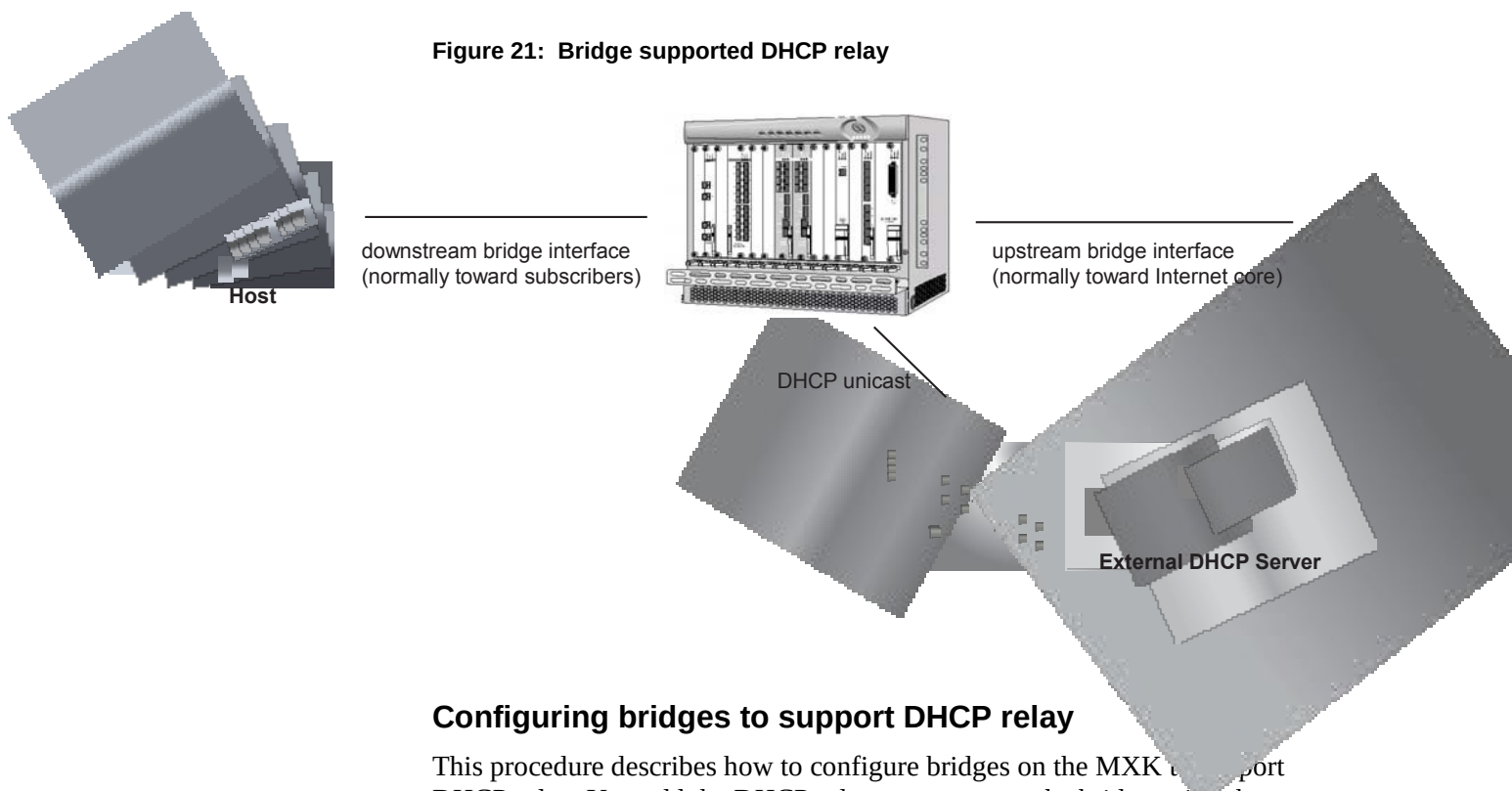
Bridge configuration with DHCP relay

The MXK enables bridges to be configured as DHCP relay agents. All DHCP messages on the bridge will have Option 82 information inserted to be passed up through an IP interface to an external DHCP server.

The MXK supports primary and alternate DHCP servers, see [DHCP relay on page 323](#).

[Figure 21](#) illustrates the traffic flow when the MXK is configured with a bridge to support DHCP relay.

Figure 21: Bridge supported DHCP relay



Configuring bridges to support DHCP relay

This procedure describes how to configure bridges on the MXK to support DHCP relay. You add the DHCP relay as you create the bridge using the **bridge add** command by adding the **dhcp-relay** rule.

Before you add DHCP relay you should have an IP interface on the MXK with a route available to the DHCP server.

There is a mechanism for add

Once the above elements are configured, to configure bridge support use the **dhcp-relay add** command.

- 1 To configure support for DHCP relay on a bridge use the **dhcp-relay add** command which uses the **subnetgroup** parameter as an identifier:

```
dhcp-relay add [<subnetgroup>] <ip-address> NULL
```

The **subnetgroup** parameter is the index identifier of the dhcp-server subnet.

The **ip-address** parameter is the address of the external DHCP server.

For DHCP relay on bridges you add the **NULL** parameter

- 2 Add the **dhcp-relay** rule using the rule add command which defines that the subnetgroup identifier is in the packet rule group.
- 3 Create bridge (or modify an existing bridge) to include the packet rule group.

Example DHCP relay support on a bridge

- 1 Configure DHCP relay support on the bridge using **dhcp-relay add**.

```
zSH> dhcp-relay add 20 11.1.1.1 NULL
Operation completed successfully.
This DHCP Relay Agent is available only for bridged connections; Routed
interfaces will not be able to use it.
Created DHCP Relay Agent number 20
```

- 2 Add the dhcp-relay rule to the IP packet rule group.

```
zSH> rule add bridgedhcprelay 10/1 20
Created packet-rule-record 10/1 (bridgedhcprelay)
```

- 3 Create bridge and include IP packet rule group.

```
zSH> bridge add 1-13-10-0/eth downlink vlan 700 ipktrule 10
Adding bridge on 1-13-10-0/eth
Created bridge-interface-record 1-13-10-0-eth/bridge
```

You can verify the information in the profiles:

```
zSH> get dhcp-server-subnet 20
dhcp-server-subnet 20
network: -----> {0.0.0.0}
netmask: -----> {0.0.0.0}
domain: -----> {0}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: -----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {0.0.0.0}
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {20} dhcp server subnet
```

```

stickyaddr: -----> {enable}
external-server: -----> {11.1.1.1} dhcp server address
external-server-alt: ---> {0.0.0.0}

```

- Verify the **dhcp-server-subnet** subnet group.
- Verify the rule exists (also a good way to find the group number:

```

zSH> rule show
Group/Member                                     Type Value(s)
-----
          10/1                                bridgedhcprelay 20
1 record(s) found

```

- Verify the **packet-rule-record** links to the DHCP server subnet group:

```

zSH> get packet-rule-record 10/1
packet-rule-record 10/1
packetRuleType: ---> {bridgedhcprelay}
packetRuleValue: --> {20}
packetRuleValue2: -> {}
packetRuleValue3: -> {}
packetRuleValue4: -> {}
packetRuleValue5: -> {}

```

- Verify the **bridge-interface-record** contains the packet rule group:

```

zSH> get bridge-interface-record-13-10-0-eth/bridge
bridge-interface-record 1-13-10-0-eth/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {700}
stripAndInsert: -----> {true}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {10} packet rule group
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}

```

```

isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

Shaping Traffic: Class of Service Queuing

Class of Service (CoS) queuing controls traffic to optimize or guarantee performance. This shaping of traffic generally exists to increase bandwidth so you can get more throughput to a device, or to decrease latency, so you do not have jitter in time sensitive data streams as in voice or video.

Congestion happens for various reasons. If you have a higher bandwidth line feeding into a smaller bandwidth line, or if you have multiple similar size lines feeding into a single line. Both of these can be considered feeding too much data (a big pipe) into a small pipe.

Queuing defines which VLAN will be able to use how much of the physical interface.

The MXK supports setting CoS values in Ethernet VLAN headers for bridged packets. This service enables you to assign a service level or CoS to an Ethernet VLAN interface that is transported across a uplink, intralink, or downlinked tagged bridge. The configured CoS level specifies the packet priority and queuing methods used to transport the packet through the Ethernet network. The MXK sets and preserves the CoS settings to ensure these settings are passed to other Ethernet devices in the network for QoS processing.

CoS values range from 0 — 7, with the lowest priority being 0 and the highest priority 7. The MXK supports eight queues per physical interface meaning that frames with a 0 CoS value are put into queue number 0; frames with a 1 CoS value are put into queue number 1, and so forth.

These are strict priority queues which mean that everything is cleared out of the high priority queue first. Only after that queue is empty is the next queue serviced. Since these are strict priority queues it is possible that the lower priority queues may get overloaded while the higher priority queues are being cleared.

Frames which require the highest throughput or are sensitive to latency (the amount of time between received packets) should be in higher priority queues. Since queuing is relative to the type of traffic, the priority settings depend on the type of traffic. Normally video and voice are more sensitive to throughput and latency issues.

Where CoS queuing takes place is dependent on the cards involved. GPON and Active Ethernet cards have queuing performed on the line card. For ADSL the queuing takes place on the uplink card.

Configuring Class of Service

The following parameters in the bridge interface record are used for Ethernet COS support.

Table 5: COS parameters in the bridge interface record profile

Parameter	Description
vlanIdCOS	Specifies the value loaded into the COS field of the VLAN header when an untagged packet received on this interface is tagged (VLAN ID inserted) for bridging. Value range is 0 to 7. Default is 0.
outgoingCOSOption	Specifies whether to insert the VLAN COS bits on packets bridged through this interface. Values: Disable Leave any existing COS values unchanged. This is the default value. All Replace the current COS values in all VLAN headers in tagged and untagged packets originating and transported through this device.
outgoingCOSValue	For outgoing tagged packets, specifies the value used to overwrite any existing COS value in the VLAN header. Value range is 0 to 7. Default is 0.

To display the **bridge-interface- record** profile, enter the **show bridge-interface-record** command.

```

zSH> show bridge-interface-record
vpi:-----> {0}
vci:-----> {0}
vlanId:-----> {0 - 2147483647}
stripAndInsert:-----> false true
customARP:-----> false true
filterBroadcast:-----> false true
learnIp:-----> false true
learnUnicast:-----> false true
maxUnicast:-----> {0 - 2147483647}
learnMulticast:-----> false true
forwardToUnicast:-----> false true
forwardToMulticast:-----> false true
forwardToDefault:-----> false true
bridgeIfCustomDHCP:-----> false true
bridgeIfConfigGroupIndex:-> {0 - 2147483647}
vlanIdCOS:-----> {0 - 7}
outgoingCOSOption:-----> disable all
outgoingCOSValue:-----> {0 - 7}

```

Adding a bridge with a CoS value

This example adds interface *1-1-1-0/adsl* with a *vlanIDCOS* value of 7. This value is inserted into the priority field of the VLAN header when an untagged packet received on this interface is tagged (VLAN ID inserted) for bridging.

```
bridge add 1-1-1-0/adsl downlink vlan 100 tagged COS 7
```

This example adds interface *1-1-1-0/adsl* with a *vlanIDCOS* value of 7 and enables the overwriting of the VLAN ID in all outgoing packets with the value of 7.

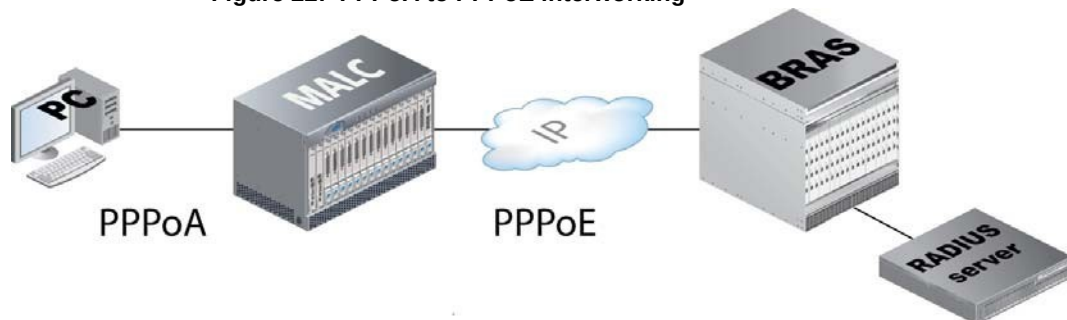
```
bridge add 1-1-1-0/adsl downlink vlan 100 tagged COS 7 outCOS all 7
```

PPPoA - PPPoE interworking

The MALC and MXK support PPPoA to PPPoE interworking for connections to a Broadband Remote Access Server (BRAS) using a PPP tunnel. Upon detecting PPPoA traffic, the device initiates a PPPoE session with the Broadband Remote Access Server (BRAS). PPP traffic between the CPE and the BRAS is tunneled over this PPPoE session. The device autosenses the type of PPPoA encapsulation as either VCMUX or LLC.

An inactivity time-out occurs when a lack of activity is detected on the PPPoA connection for 30-80 seconds, while upstream PPPoE packets are received. When this occurs, the PPPoE session is terminated.

Figure 22: PPPoA to PPPoE interworking



Enabling PPPoA to PPPoE interworking

PPPoA – PPPoE interworking is added by enabling PPPoA on an ADSL downlink bridge.

The **bridge add** command supports enabling PPPoA interworking from the CLI. This example creates a downlink bridge on the interface *1-9-24-0/adsl* with VLAN 500 and enables the PPPoA to PPPoE feature.

```
zSH> bridge add 1-9-24-0/adsl vc 0/35 td 1 downlink vlan 500 pppoa
Adding bridge on 1-9-24-0/adsl
Created bridge-interface-record 1-9-24-0-adsl-0-35/bridge
```

This command automatically updates the **bridge-interface record** profile.



Note: The following message may appear if the CPE device is not properly configured for PPPoA connections.

```
FEB 01 15:59:22: error   : 1/1/9   : bridge:
_afsmChkRcvEncaps(): l=1811: tNetTask:
AFSM-6313: port 1-9-24-0-adsl-0-35 misconfigured
for PPPoA
```

Verifying PPPoA – PPPoE interworking

1 Verify the PPPoA parameter in the **bridge-interface-record**

```
zSH> get bridge-interface-record 1-9-24-0-adsl-0-35/bridge
bridge-interface-record 1-9-24-0-adsl-0-35/bridge
vpi: -----> {0}
vci: -----> {35}
vlanId: -----> {500}
stripAndInsert: -----> {true}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {true}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}
```

2 Use the **bridge show** command to display the state of the PPPoA session.

When the PPPoA port status is UP, the BRAS MAC address and PPPoE session ID are also displayed.

PPPoA port states are:

- PENDING (PND)

The bridge port has not yet bound with the driver during initialization. This state is for all bridges. A bridge cannot transition back to this state.

- READY (RDY)

Waiting for PPPoA packet to initiate PPPoE discovery.

- UP

The PPPoA port is active. The BRAS MAC address and PPPoE session ID will also be displayed.

- DOWN (DWN)

The PPPoA port is down

- DISCVRY (DSC)

PPPoE discovery initiated. Waiting for session ID to be obtained.

PPPoA port is pending.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
upl	Tagged	ethernet2-0/bridge	UP	S	Global default
poa	500	1-6-27-0-adsl-0-35/bridge	PND		
poa	500	1-6-19-0-adsl-0-35/bridge	PND		

PPPoA port is ready.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
upl	Tagged	ethernet2-0/bridge	UP	S	Global default
poa	500	1-6-27-0-adsl-0-35/bridge	RDY		
poa	500	1-6-19-0-adsl-0-35/bridge	RDY		

PPPoA port is up.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
upl	Tagged	ethernet2-0/bridge	UP	S	Global default
poa	500	1-6-27-0-adsl-0-35/bridge	UP	A	00:19:aa:3b:83:30 8587
poa	500	1-6-19-0-adsl-0-35/bridge	UP	A	00:19:aa:3b:83:30 8586

Bridges with packet rule records

This section explains how to configure **packet-rule-record** filters and includes:

- [Mechanism for multiple interface ingress filters, page240](#)
- [Destination MAC swapping, page243](#)
- [Bandwidth limiting by port and service, page245](#)
- [Bridge configuration with DHCP relay, page232](#)
- [DHCP on bridge packet rules \(DHCP relay, Option 82, and Forbid OUI\), page253](#)
- [PPPoE with intermediate agent, page256](#)

Mechanism for multiple interface ingress filters

The SLMS CLI architecture has a mechanism for adding multiple filters for ingress interfaces by grouping **packet-rule-record(s)**.

In the bridge interface record you configure the ingress interface packet rule group index. Multiple bridges may use the same ingress interface packet rule group index.



You can add multiple filters with the **rule add** command by supplying both the group index and the member index when you add a rule. The **bridge-interface-record** accesses rules by the group index number.

```
rule add <groupIndex/memberIndex> <packetRuleType>
<packetRuleValue...packetRuleValue2>
```

The **packetRuleValue** options depend on the **packetRuleType** selected. For example, when using **bridgeinsertoption82**, you have two **packetRuleValues**, one for circuit ID and one for remote ID.

```
zSH> rule add bridgeinsertoption82 10/2 "circuitIDExample"
```

```
zSH> rule add bridgeinsertoption82 10/2 "circuitIDExample" "remoteIDExample"
```

The **bridge add** command then has a parameter which refers to the group with the **ipktrule** or the **epktrule** variables. Entering **ipktrule** adds the filter on the bridge ingress and **epktrule** add the filter on the bridge egress. Filters are asymmetrical, meaning that the same filter can be applied to the ingress and the egress of the bridge using different values.

For example:

```
zSH> bridge add 1-13-1-0/eth vlan 777 ipktrule 2
```

Packet rules are used for the following features and their options:

- [Destination MAC swapping, page243](#)
- [Bandwidth limiting by port and service, page245](#)
- [DHCP on bridge packet rules \(DHCP relay, Option 82, and Forbid OUI\), page253](#)
- [PPPoE with intermediate agent, page256](#)

Configure packet-rule-records

Adding a packet-rule-record and packet rule group

- 1 Use the **rule add** command to add the rule giving a group index and member index, rule type and the parameters which that rule type requires.

The first example creates a member for the IP packet group with the index of “2”. The **dstmacswappingstatic** rule shown requires a parameter which is a MAC address. Entering *ipktrule* will enter the rule on the ingress of the bridge.

```
zSH> rule add dstmacswapstatic 2/1 08:00:20:bc:8b:8c
Created packet-rule-record 2/1 (dstmacswapstatic)
```

- 2 Create the bridge and include the IP packet rule group

```
zSH> bridge add 1-13-5-0/eth downlink vlan 777 ipktrule 2
Adding bridge on 1-13-5-0/eth
Created bridge-interface-record 1-13-5-0-eth/bridge
```

Deleting a packet rule

Use the **rule delete** command to delete the rule.

```
zSH> rule delete 2/1
packet-rule-record 2/1 Delete complete
```

Verifying packet rule groups

Use the **rule show** command to display the rules.

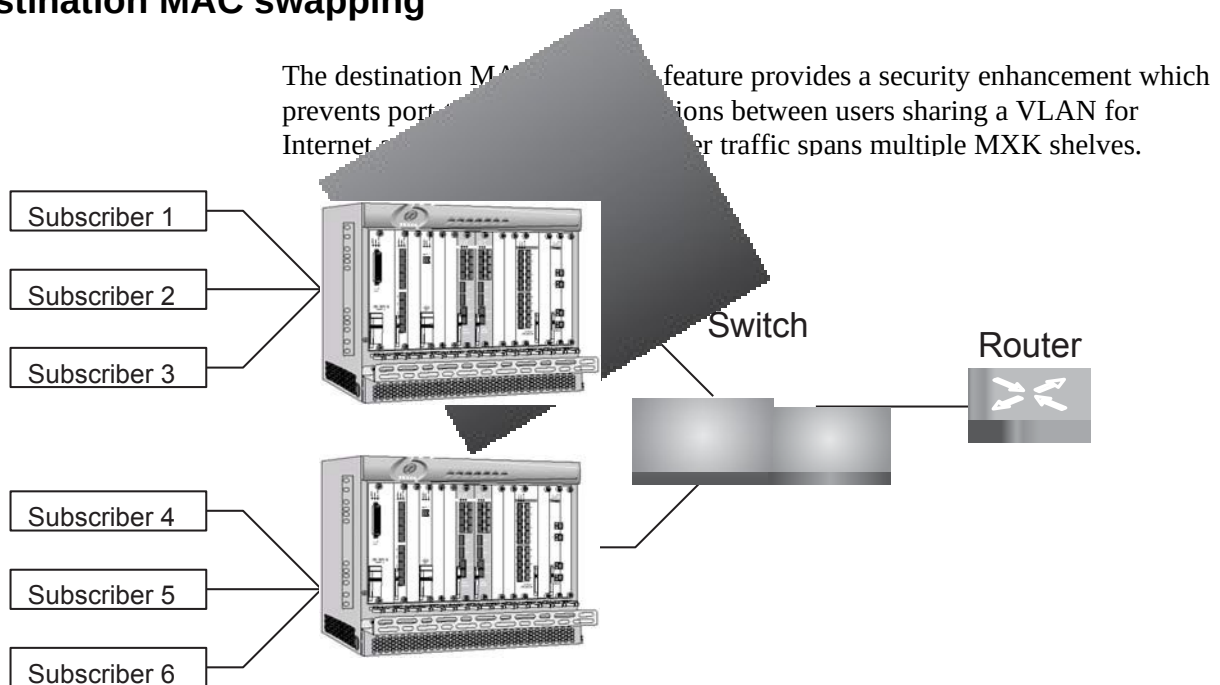
```
zSH> rule show
Group/Member                                     Type Value(s)
```

```

-----
2/1          dstmacswapstatic 08:00:20:bc:8b:8c
4/1          ratelimitdiscard 1300 120000 130000
10/1         bridgedhcprelay 20
3 record(s) found

```

Destination MAC swapping



When enabled, this feature modifies the destination MAC address portion of unicast frames (Ethernet frames not using a multicast or broadcast destination MAC) that traverse the MXK so that the destination MAC is changed to the MAC address of the next-hop router in the access network. This address modification ensures that all frames in the access network are forwarded to the access router regardless of how the frame originated. Broadcast, multicast, and Ethernet frames with a destination MAC address of the next hop router are forwarded without MAC swapping.

The MXK retrieves the MAC address of the next hop router to correctly swap into unicast frames through dynamically snooping DHCP ACK messages or a static user-specified entry.

- Dynamically snooping DHCP ACK messages

The MXK snoops DHCP ACK messages received on the bridge interface that is configured as the default (VLAN or default bridge). The source MAC address from this frame is swapped into for frames received on interfaces configured for destination MAC swapping. This address is stored in the database and persists across reboots. When a new DHCP ACK message is received in the same VLAN, its source is checked, and if different, the newer MAC address is used.

This option requires that DHCP server services are used in the network and that the next hop router is the default router between the MXK and the DHCP server.

- Static user-specified entry

The MXK inserts the user-specified valid 6-byte hexadecimal MAC address into unicast frames not matching the static entry.



Note: Destination MAC swapping is only supported on the uplink cards on the MXK.

Configuring destination MAC swapping

Use the **rule add** command to create either the dynamic or static destination MAC swapping rule:

```
rule add <dstmacswapdynamic|dstmacswapstatic>
<groupindex/Memberindex> <MAC address>
```

The rule for dynamic MAC swapping does not have a parameter. The rule for static MAC swapping requires a parameter, the MAC address to match.

```
rule add dstmacswapdynamic groupindex/Memberindex
```

```
rule add dstmacswapstatic groupindex/Memberindex macaddress
```

dstmacswapdynamic or dstmacswapstatic

MAC addresses of the net hop router used to correctly swap into unicast frames through either dynamically snooping DHCP ACK messages or a static user-specifies entry.

Syntax **dstmacswapdynamic or dstmacswapstatic**

Options **dstmacswapdynamic**

Dynamic MAC swapping reads the destination MAC address from the default VLAN on the uplink to swap into the packet, so you just need to define which uplink bridge interface to associate with the rule.

dstmacswapstatic

Static MAC swapping requires a MAC address to be swapped into the packet which you must supply.

Example 1 For dynamic MAC swapping:

```
zSH> rule add dstmacswapdynamic 1/1
Created packet-rule-record 1/1 (dstmacswapdynamic)
```

Example 2 For static MAC swapping:

```
zSH> rule add dstmacswapstatic 2/1 08:00:20:bc:8b:8c
Created packet-rule-record 2/1 (dstmacswapstatic)
```

Bandwidth limiting by port and service

This section describes these topics:

- [Rate limiting overview, page245](#)
- [Color blind rate limiting, page248](#)
- [Configure color blind policing, page248](#)
- [Color aware rate limiting, page252](#)
- [Configure color aware policing, page252](#)

Rate limiting overview

Rate limiting is typically used when a service provider needs to provide customer services with limited bandwidth and needs to create a priority for which type of packets — data, voice, or video — have priority when there is bandwidth contention. In other words, a service provider may need to ensure that video traffic gets to the user at the expense of data or voice traffic.

You use rate limiting to control the rate of traffic sent or received on the ingress or the egress of both the logical port or the physical port on the MXK. Traffic that is less than or equal to the specified rate is sent and traffic that exceeds the rate is dropped or delayed.

Rate limiting is supported on the ingress and egress of SHDSL-EFM, GPON, Ethernet, and VDSL cards. ADSL cards use traffic descriptors for modulating traffic.

After configuring an interface with rate limiting, the traffic rate is monitored and metered to verify conformity with an established contract.

Non-conforming traffic is discarded, while conforming traffic passes through the interface without any changes. The MXK follows RFC 2697 for rate limiting on both the ingress and egress of the interface.

The two modes of rate limiting are:

- Color blind

Rate limiting is performed on the interface without using the frame's Class of Service (CoS) by assuming that all packets of a flow are “uncolored” and are treated equally.

Color blind mode is most commonly used for a single service per VLAN.

- Color aware

Rate limiting observes that the incoming packet flow is colored and each packet is marked green, yellow, or red to signify if a packet has high, medium, or low priority. The color field maps to the priority CoS value in tagged packets and the IP precedence TOS value in untagged packets.

Color aware mode is most commonly used for multiple services on a single VLAN to ensure that the higher priority packets get through if there is bandwidth contention.

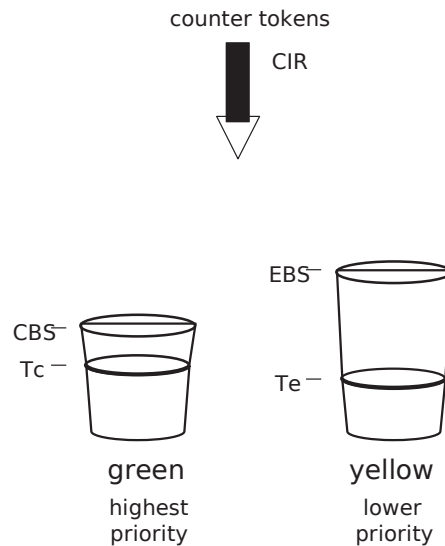


Note: Color values are not supported on egress ports.

The single rate color marker scheme from RFC 2697 uses a counter to gauge capacity on the line by counting tokens. The counters are used to determine which packets get dropped. The idea is that the green bucket fills up faster than the yellow buckets.

There are three parameters which determine which packets are dropped — a rate, Committed Information Rate (CIR) which supplies tokens to be counted, and two buckets, Committed Burst Size (CBS) and Excess Burst Size (EBS), which provide two levels of priority. [Figure 23](#) describes a single rate counter scheme.

Figure 23: Single rate counter scheme



CIR is the rate which determines how quickly the token buckets fill up. Both buckets start full. It is important to understand that this is not a buffering scheme as incoming packets are not queued up for later delivery.

For every CIR increment the buckets are filled.

```

if Tc < CBS
  then
    increment Tc
  else if Te < EBS
    then
      increment Te
    else
      do nothing (do not increment either because
                  they are both full)

```

The green bucket will fill first and faster if it is not full because the yellow bucket will not increment until $Tc \geq CBS$.

There are rules about how the green bucket size (CBS) and yellow bucket size (EBS) should be configured. At least one of CBS or EBS should be greater than zero. Also at least one of CBS or EBS should be greater than the largest expected packet in the incoming stream, as packets which are larger than both CBS or EBS will be dropped. Normally you would have CBS greater than EBS, so packets that do not go because there are not enough green tokens will go because there are enough yellow tokens.

With color blind rate limiting the size of the incoming packet determines whether the packet will go. If there are enough tokens in green or yellow it will go. Tokens matching the size of the packet will be decremented from the appropriate bucket. If there are packets which are larger than the amount of tokens in either bucket, those packets are dropped. Packets which are larger than either bucket size when full are dropped.

```
if incoming packet smaller than Tc
then
    decrement Tc by size of packet
    send packet
else if packet smaller than Te
then
    decrement Te by size of packet
    send packet
else
    drop packet
```

With color aware rate limiting, it is assumed that the packets are being marked by an upstream device. Packets which are marked red are dropped. Packets which are marked yellow are best effort and green are highest priority and should have the lowest chance of the packet being dropped. The behavior depends on the configuring of the CBS and EBS parameters.



Note: The default values for CBS and EBS are good for most situations. Only advanced users should change these values.

With color aware rate limiting the size and the color determine whether the packet will be dropped.

```
if incoming packet is green AND is smaller than Tc
then
    decrement Tc by size of packet
    send packet
else if packet is green or yellow AND is smaller than Te
then
    decrement Te by size of packet
    send packet
else
    drop packet
```

So all red packets are dropped. Normally the upstream marking device will mark packets red which are too large.

Color blind rate limiting

Color blind rate limiting is usually set when one service is supplied per VLAN. The rate limit, CIR, is set in kilobytes per second. For any rate above the set CIR, packets will drop.

For example, in [Figure24](#), you would use the color blind method to set VLAN 100 to drop packets when the rate exceeds 5 Mbps, VLAN 200 to drop packets when the rate exceeds 3 Mbps, and VLAN 200 to drop packets when the rate exceeds 6 Mbps.

Figure 24: One service per VLAN on an interface

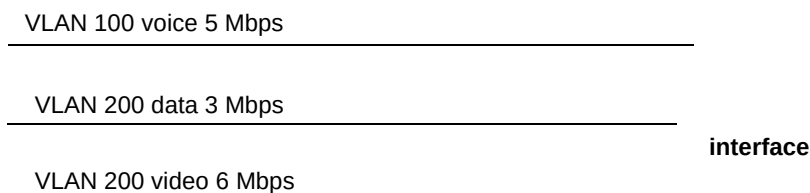


Table 6: Definition of rate limiting controls

Acronym	Definition	Rate	Description
CIR	Committed Information Rate	kbps	The average rate guaranteed for a virtual circuit. If the actual rate goes above the CIR the packets will be dropped.
CBS	Committed Burst Size	bps	The maximum data rate which can be carried under normal conditions. This rate is greater than the CIR, but less than the EBS.
EBS	Excess Burst Size	bps	The maximum data rate that the circuit will attempt to carry.



Note: The default values for CBS and EBS are good for most situations. Only advanced users should change these values.

Configure color blind policing

The **rule add ratelimitdiscard** command sets the rate above which packets will be dropped.

(value1 is CIR, value2 is CBS, value3 is EBS)

```
rule add ratelimitdiscard <groupIndex/memberIndex> rate <rate> [cbs <value>] [ebs <value>]
```

Service providers can use two color blink rate limiting filters to service subscribers that will allow higher bandwidth coming from the network

through the MXK to the subscriber and lower bandwidth leaving the subscriber through the MXK to the network.

Configuring color blind policing filters for the ingress and the egress of a bridge

This example describes setting the rate above which packets are dropped on an Ethernet downlink bridge.

- 1 Create the rule for the ingress from the subscriber to the MXK.

```
zSH> rule add ratelimitdiscard 3/1 rate 1300
Created packet-rule-record 3/1 (ratelimitdiscard)
```

- 2 Create the rule for the egress from the MXK to the subscriber.

```
zSH> rule add ratelimitdiscard 4/1 rate 6000
Created packet-rule-record 4/1 (ratelimitdiscard)
```

- 3 View the rules.

```
zSH> rule show
Group/Member                                     Type Value(s)
-----
          3/1                                ratelimitdiscard 1300 120000 130000
          4/1                                ratelimitdiscard 6000 120000 130000
```

To view just the **ratelimitdiscard** rules enter:

```
zSH> rule show ratelimitdiscard
Group/Member                                     Type Value(s)
-----
          3/1                                ratelimitdiscard 1300 120000 130000
          4/1                                ratelimitdiscard 6000 120000 130000
2 record(s) found
```

- 4 Apply the rule to the egress of a downlink Ethernet bridge from the MXK to the subscriber.

```
zSH> bridge add 1-13-8-0/eth downlink vlan 555 tagged ipktrule 3/1
Adding bridge on 1-13-8-0/eth
Created bridge-interface-record 1-13-8-0-eth-555/bridge
```

To apply a filter with the **bridge add** command to the egress, you would use **epktrule**.

The group index number is applied to the **bridgeIfIngressPacketRuleGoupIndex** parameter in the **bridge-interface-record**.

```
zSH> get bridge-interface-record 1-13-8-0-eth-555/bridge
bridge-interface-record 1-13-8-0-eth-555/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {555}
```

```

stripAndInsert: -----> {false}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {3}  rule applied to the ingress
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

- 5 Apply rule to the egress of the downlink bridge by entering the group index number to the **bridgeIfEgressPacketRuleGroupIndex** parameter in the **bridge-interface-record** profile of the downlink bridge.

The way to apply a second rule to the same bridge is by updating the **bridge-interface-record**.

```

zSH> update bridge-interface-record 1-13-8-0-eth-555/bridge
bridge-interface-record 1-13-8-0-eth-555/bridge
Please provide the following: [q]uit.
vpi: -----> {0}:
vci: -----> {0}:
vlanId: -----> {555}:
stripAndInsert: -----> {false}:
customARP: -----> {false}:
filterBroadcast: -----> {false}:
learnIp: -----> {true}:
learnUnicast: -----> {true}:
maxUnicast: -----> {5}:
learnMulticast: -----> {true}:
forwardToUnicast: -----> {false}:
forwardToMulticast: -----> {false}:
forwardToDefault: -----> {true}:

```

```

bridgeIfCustomDHCP: -----> {false}:
bridgeIfIngressPacketRuleGroupIndex: -> {4}:
vlanIdCOS: -----> {0}:
outgoingCOSOption: -----> {disable}:
outgoingCOSValue: -----> {0}:
s-tagTPID: -----> {0x8100}:
s-tagId: -----> {0}:
s-tagStripAndInsert: -----> {true}:
s-tagOutgoingCOSOption: -----> {s-tagdisable}:
s-tagIdCOS: -----> {0}:
s-tagOutgoingCOSValue: -----> {0}:
mcastControlList: -----> {}:
maxVideoStreams: -----> {0}:
isPPPoA: -----> {false}:
floodUnknown: -----> {false}:
floodMulticast: -----> {false}:
bridgeIfEgressPacketRuleGroupIndex: --> {0}: 4 add the second filter
bridgeIfTableBasedFilter: -----> {NONE(0)}:
bridgeIfDhcpLearn: -----> {NONE(0)}:
mvrVlan: -----> {0}
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

6 Verify the packet rules.

```

zSH> get bridge-interface-record 1-13-8-0-eth-555/bridge
bridge-interface-record 1-13-8-0-eth-555/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {555}
stripAndInsert: -----> {false}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {3}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}

```

```
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {4}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}
```

Color aware rate limiting

 **Note:** Not commonly used except when performing advanced configurations.

Color aware bandwidth limiting is usually used when multiple services with different priorities are offered on a single VLAN. The colors green, yellow, and red are used for metering traffic and the colors correspond to COS values that range from 0-7. You can set which colors correspond to which COS value.

Color Aware Policing is based on the idea that upstream devices are policing and marking frames based on a set of rules. A green packet is well behaved. A yellow packet has misbehaved at some point so if there is a bandwidth congestion it should be dropped before a green frame. A red packet has violated a rule and should be dropped. This means that green packets are serviced first, then if there is enough room, the yellow packets are serviced. Red packets are always dropped.

Table7 shows the default mapping of CoS value to color.

Table 7: Default Color to CoS values

CoS value	Color
7	green
6	green
5	green
4	green
3yellow	
2yellow	
1yellow	
0yellow	

Configure color aware policing

The **rule add colorawareratelimitdiscard** command sets the color priority and the rate above which packets will be dropped.

The high-priority and low-priority parameters allow to designate which values on the scale will be treated as green, yellow and red. If high-priority is set to 5 and the low-priority set to 3, the CoS value to color table will change so that 7, 6, and 5 are green; 4 and 3 will be yellow; and 2, 1 and 0 will be dropped

```
rule add colorawareratelimitdiscard <groupIndex/memberIndex> rate <rate> [cbs
<value>] [ebs <value>] [hi-priority <value>] [low-priority <value>]
```

For example,

```
zSH> rule add colorawareratelimitdiscard 5/1 rate 1300 hi-priority
Created packet-rule-record 5/1 (colorawareratelimitdiscard
```

Value1 is CIR, value2 is CBS, value3 is EBS, value4 is hi-priority, value5 is low-priority. In this instance the hi-priority and low-priority are set at the defaults as shown in [Table7](#).

To view just the **colorawareratelimitdiscard** rules just created enter:

```
zSH> rule show colorawareratelimitdiscard
Group/Member                                Type Value(s)
-----
          5/1                                colorawareratelimitdiscard 1300 120000 130000 4 0
1 record(s) found
```

DHCP on bridge packet rules (DHCP relay, Option 82, and Forbid OUI)

The MXK supports multiple **packet-rule-record(s)** via a grouping mechanism so an open-ended number of filter settings can be configured for a bridge interface (see the example in [Bridge configuration with DHCP relay on page232](#)). The same filter settings can also be easily applied to multiple bridge interfaces.

In uplink and downlink bridges **packet-rule-record(s)** are typically assigned to bridge configuration groups on downlink bridge interfaces.

Add the DHCP packet rule options using the **rule add** command to specify the packet rule option and which **packet-rule-record** group.

General case of adding DHCP packet rules

- 1 Use the **rule add** command with the appropriate **packetRuleType** and **packetRuleValue(s)** and packet rule group.
- 2 Create bridge (or modify an existing bridge) to include the packet rule group.

The **bridge-interface-record** contains a reference to the **packet-rule-record**. Multiple **packet-rule-record(s)** may be put into a packet rule group by using a *m/n* identifier, where *m* is the identifier of the group and *n* is the identifier for the specific **packet-rule-record**.

The **packetRuleType** and, where appropriate a **packetRuleValue** parameter or parameters specify the variety of filters to be applied to the interface.

```
zSH> rule add <packetRuleType> <groupIndex/memberIndex> <packetRuleValue1>
<packetRuleValue2> ...
```

See the following DHCP packet rule records for appropriate **packetRuleType** and **packetRuleValues** for the **rule add** command:

- bridgeddhcprelay:

packetRuleValue contains the DHCP subnet group ID. If only the DHCP relay option is used, option82 information is displayed in hex format as *slot port shelf vlan*. See [Configuring bridges to support DHCP relay, page232](#).

```
zSH> dhcp-relay add 20 11.1.1.1 NULL
zSH> rule add bridgedhcprelay 10/1 20
```

- bridgeinsertoption82:

You can define textual values for two items of textual information: circuit ID and remote ID.

If the first value is set it is taken as a literal text string to be used as the suboption 1 field in the DHCP packet. If it is not set a text string identifying the box and interface which received the packet is used. If the second value is set it is taken as a literal text string to be used as the suboption 2 field in the DHCP packet. If it is not set no suboption2 is provided.

Use of this feature will usually require a distinct rule group for each interface since the circuit and remote Id values associated with suboptions 1 and 2 are distinct for each interface.

When acting as a DHCP relay agent, the MXK includes option 82 to identify the requesting client to the DHCP server. There are two sub-options for DHCP option 82 insert — Circuit ID and Remote ID. Both of these fields are text fields, though they were designed to carry specific information. It is up to your implementation plans to define how to use the option 82 inserts.

Circuit ID is meant to provide information about the circuit which the request came in on. It is normally the port and interface information.

RFC 3046 describes possible uses of the Circuit ID field:

- Router interface number
- Remote Access Server port number
- Frame Relay DLCI
- ATM virtual circuit number
- Cable Data virtual circuit number

Remote ID is meant to provide information about the remote host end of the circuit, however in practice the sub-option usually contains information about the relay agent.

RFC 3046 describes possible uses of the Remote ID field:

- a "caller ID" telephone number for dial-up connection
- a "user name" prompted for by a Remote Access Server
- a remote caller ATM address
- a "modem ID" of a cable data modem
- the remote IP address of a point-to-point link
- a remote X.25 address for X.25 connections

Since both fields support textual insertions on the MXK, please research RFC 3046 for further details regarding field format.

To specify neither circuit ID or remote ID value:

```
zSH> rule add bridgeinsertoption82 1/1 ""
```

To specify only the first circuit ID value:

```
zSH> rule add bridgeinsertoption82 1/1 CircuitIdText
```

To specify only the second remote ID value:

```
zSH> rule add bridgeinsertoption82 1/1 "" RemoteIdText
```

To specify both values:

```
zSH> rule add bridgeinsertoption82 1/1 "CircuitIdText" "RemoteIdText"
```

- bridgeinsertpppoevendortag

packetRuleValue contains optional identification string that is converted to TR101 compliant data.

```
zSH> rule add bridgeinsertpppoevendortag 1/1
```

Filtering access based on organizational unique identifier (OUI)

When using the bridgeforbidoui option for a packet rule, you provide the first three bytes of the MAC address which are used to identify vendors. These three bytes are known as the Organizational Unique Identifier (OUI)/

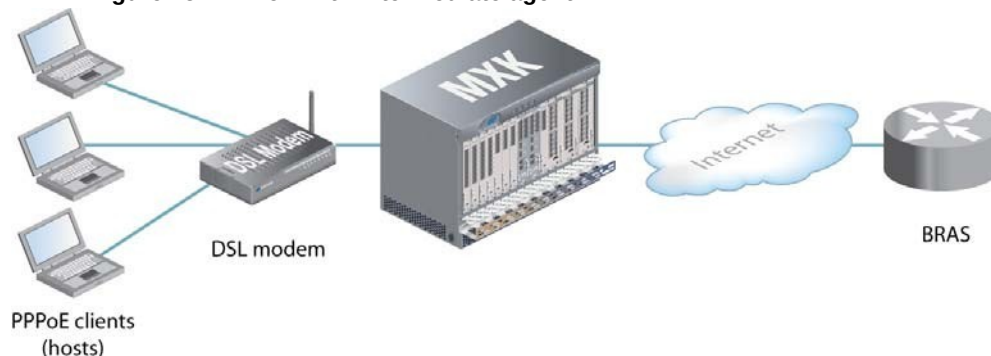
```
zSH> rule add bridgeforbidoui 1/1 AA:BB:CC
```

Packets from a device with a MAC address which begins with "AA:BB:CC" the hexadecimal vendor code (OUI — Organizational Unique Identifier) will be blocked.

PPPoE with intermediate agent

The MXK is capable of being an intermediate agent in a PPPoE (point-to-point protocol over Ethernet) scenario. In a PPPoE scenario, PPPoE clients initiate the connection process and need to learn the Ethernet address of the remote peer and establish a unique session ID to establish a connection.

Figure 25: PPPoE with intermediate agent



In the discovery process the PPPoE client (host) broadcasts a request by transmitting PPPoE Active Discovery Initiation (PADI) packet. When one or more responses are received by the host (the responses include the address of the Broadband Remote Access Server (BRAS)), the host then sends a unicast PPPoE Active Discovery Request (PADR) packet.

The MXK supports inserting port information into PPPoE packets that transmit a MXK bridge interface. When the MXK receives a PPPoE Active Discovery Initiation (PADI) packet or a PPPoE Active Discovery Request (PADR) packet, the MXK can be configured to insert a customized string along with default port/slot identification into the vendor-specific portion of the PPPoE packet.

The vendor-specific tag containing the customized identification string can be used to identify a circuit and send this value to a RADIUS (remote access dial-in service) server or network server. The customized string could also be used for record keeping.

The customized identification string can be 0 to 48 characters. The inserted information is TR-101 compliant and formatted as:

```
<customstring> eth slot/port[[:stagID]:vlan-tag]
```

The slot/port values identify the ingress slot/port on the MXK where the packet was received. If the packet is tagged with a VLAN tag, the VLAN tag is also added to the packet on ingress. If the packet is tagged with a SLAN tag, the SLAN tag is also added to the packet on ingress.

```
zSH> rule add bridgeinsertpppoevendortag 1/1
```

Examples of inserted tags:

- Untagged packet no customized string from slot 5 port 2

eth 5/2

- VLAN 500 tagged packet no customized string from slot 5 port 2

eth 5/2 :500

- VLAN 500 tagged, SLAN 4 tagged packet no customized string from slot 5 port 2

eth 5/2 :4 :500

- VLAN 500 tagged, SLAN 4 tagged packet with customized string of “172.42.10.5” from slot 5 port 2

172.42.10.4 eth 5/2 :4 :500

Note: For configurations with bridge intralinks or subtended SLMS devices, ensure that the PPPoE intermediate agent feature is enabled on only the subtended devices.

ADSL2+ line rate on PPPoE-IA signaling

PPP headend servers (also known as Broadband Remote Access Servers or BRAS) authenticate and manage predominately DSL customers. TR-101 defines information which is entered into the packets when creating Point to Point Protocol over Ethernet connection via an Intermediate Agent (PPPoE Intermediate Agent).

PPP sessions take place two ways on the MXK. First using PPPoE-IA, the PPP intermediate agent snoops PPPoE packets and inserts information as needed through user provisioning. The second way is when PPPoA is enabled, the MXK converts PPPoA sessions that take place with the customer CPE, and initiates a PPPoE session with the BRAS.

The PPPoE Intermediate Agent option is implemented by applying a bridge filter to the bridge-interface-record of a bridge by creating a packet-rule-record with the packetRuleType parameter set to bridgeinsertpppoevendortag and the packetRuleValue set to the customized string which identifies the circuit ID to be inserted into PADI and PADR packets.

Another requirement in TR-101 specifies the format for ADSL upstream and downstream rates and how they are added to the PPP packet Vendor Specific Tag as sub-options 0x81 and 0x82, respectively.

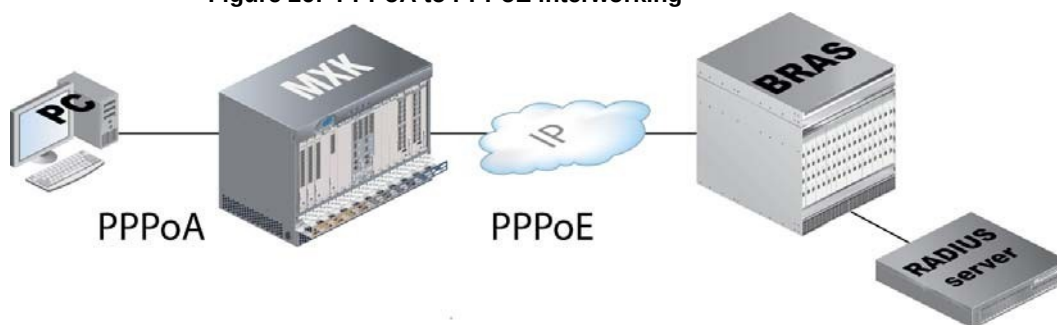
When the **bridgeinsertpppoevendortag** packet rule is entered both upstream and downstream ADSL line rate information is added into the PPP PADI and PADR packets going to the server. This line rate information gives the headend BRAS system visibility to individual line rates so that it can manage the total bandwidth being forwarded to the line. To verify the line rate information you can compare the line rate information displayed in the **dslstat** command with the line rate information received at the BRAS, or by using a packet inspection tool to confirm that the line rate data has been added to the packets.

PPPoA - PPPoE interworking

The MXK supports PPPoA to PPPoE interworking for connections to a Broadband Remote Access Server (BRAS) using a PPP tunnel. Upon detecting PPPoA traffic, the MXK initiates a PPPoE session with the Broadband Remote Access Server (BRAS). PPP traffic between the CPE and the BRAS is tunneled over this PPPoE session. The MXK autosenses the type of PPPoA encapsulation as either VCMUX or LLC.

An inactivity timeout occurs when a lack of activity is detected on the PPPoA connection for 30-80 seconds, while upstream PPPoE packets are received. When this occurs, the PPPoE session is terminated.

Figure 26: PPPoA to PPPoE interworking



Enabling PPPoA to PPPoE interworking

PPPoA – PPPoE interworking is added by enabling PPPoA on an ADSL downlink bridge.

The **bridge add** command supports enabling PPPoA interworking from the CLI. This example creates a downlink bridge on the interface *1-9-24-0/adsl* with VLAN 500 and enables the PPPoA to PPPoE feature.

```
zSH> bridge add 1-9-24-0/adsl vc 0/35 td 1 downlink vlan 500 pppoa
Adding bridge on 1-9-24-0/adsl
Created bridge-interface-record 1-9-24-0-adsl-0-35/bridge
```

This command automatically updates the **bridge-interface record** profile.



Note: The following message may appear if the CPE device is not properly configured for PPPoA connections.

```
FEB 01 15:59:22: error : 1/1/9 : bridge:
_afsmChkRcvEncaps(): l=1811: tNetTask:
AFSM-6313: port 1-9-24-0-adsl-0-35 misconfigured
for PPPoA
```

Verifying PPPoA – PPPoE interworking

1 Verify the PPPoA parameter in the **bridge-interface-record**

```

zSH> get bridge-interface-record 1-9-24-0-adsl-0-35/bridge
bridge-interface-record 1-9-24-0-adsl-0-35/bridge
vpi: -----> {0}
vci: -----> {35}
vlanId: -----> {500}
stripAndInsert: -----> {true}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {true}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

2 Use the **bridge show** command to display the state of the PPPoA session.

When the PPPoA port status is UP, the BRAS MAC address and PPPoE session ID are also displayed.

PPPoA port states are:

- PENDING (PND)

The bridge port has not yet bound with the driver during initialization. This state is for all bridges. A bridge cannot transition back to this state.

- READY (RDY)

Waiting for PPPoA packet to initiate PPPoE discovery.

- UP

The PPPoA port is active. The BRAS MAC address and PPPoE session ID will also be displayed.

- DOWN (DWN)

The PPPoA port is down

- DISCVRY (DSC)

PPPoE discovery initiated. Waiting for session ID to be obtained.

PPPoA port is pending.

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
upl Tagged     ethernet2-0/bridge                  UP  S Global default
poa           500 1-6-27-0-adsl-0-35/bridge        PND
poa           500 1-6-19-0-adsl-0-35/bridge        PND
```

PPPoA port is ready.

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
upl Tagged     ethernet2-0/bridge                  UP  S Global default
poa           500 1-6-27-0-adsl-0-35/bridge        RDY
poa           500 1-6-19-0-adsl-0-35/bridge        RDY
```

PPPoA port is up.

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
upl Tagged     ethernet2-0/bridge                  UP  S Global default
poa           500 1-6-27-0-adsl-0-35/bridge        UP  A 00:19:aa:3b:83:30 8587
poa           500 1-6-19-0-adsl-0-35/bridge        UP  A 00:19:aa:3b:83:30 8586
```

Advanced bridging topics

This section describes:

- [Bridges with IGMP, page260](#)
- [“Denial of Service” prevention, page264](#)
- [Rapid Spanning Tree Protocol \(RSTP\), page264](#)
- [Bridging differences between the MALC and MXK, page274](#)

Bridges with IGMP

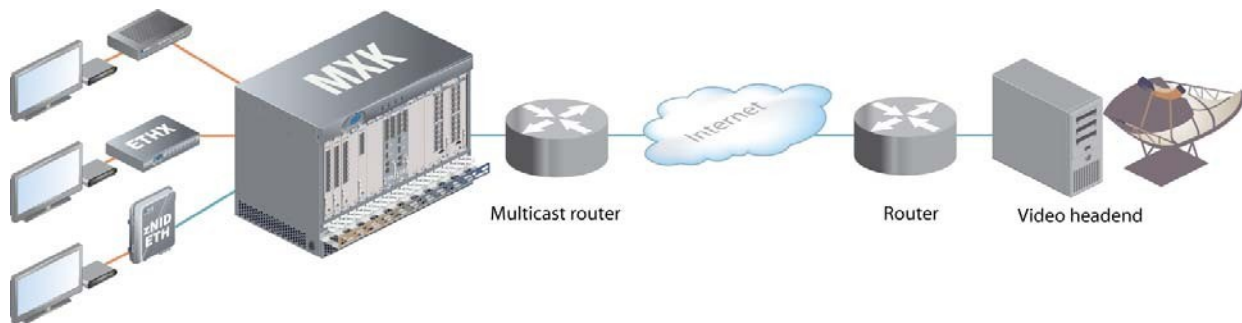
With IGMP Multicast, rather than sending frames to all devices as a broadcast which can slow down the network because it takes a lot of computation time

to duplicate packets (since this is an IP service), packets are sent to a single device and sent out only to the devices that request the service.

Video bridging on SLMS devices provides the ability to integrate video streams for multiple sources into one conduit, enabling video packets to be forwarded over a Layer 2 bridge from a host to a subscriber. As a result, the video travels from its source, or head-end device, and passes through the SLMS in a passive manner with only one video stream across the backplane, reducing bandwidth required for video packets to traverse the device.

The MXK and MALC support IGMPv1, v2, IGMP snooping and IGMP proxy reporting.

Figure 27: IGMP video



The following video bridge example describes a video bridge on an uplink GigE interface and the bridge path on that interface. The downlink bridge places an ADSL interface in shelf 1, slot 3, port 1 and assigns VCI/VPI 0/37 with traffic descriptor 1 and VLAN 800 to the downlink interface.

For the uplink bridge enter:

```
zSH> bridge add 1-a-2-0/eth uplink vlan 800
Adding bridge on 1-a-2-0/eth
Created bridge-interface-record ethernet2-800/bridge
```

For the uplink bridge path, add a bridge path and a multicast aging period and IGMP query interval.

```
zSH> bridge-path add ethernet2-800/bridge vlan 800 default mcast 90 igmptimer
30
Bridge-path added successfully
```

For the downlink bridge, add a downlink bridge and specify a maximum number of video streams and multicast control list. Members of the multicast control list must be defined to receive the video signal.

```
zSH> bridge add 1-3-1-0/adsl vc 0/37 td 1 downlink vlan 800 video
maxvideostreams 2 mcastctrl 1
Adding bridge on 1-3-1-0/adsl
Created bridge-interface-record 1-3-1-0-adsl-0-37
```

Verifying bridge settings

To verify bridge settings, use the **get bridge-interface-record** command for each bridge. This command displays the bridge settings, including the **learnMulticast** and **forwardToMulticast**.

Video bridging requires both an uplink bridge and a downlink bridge. On the uplink bridge, the **forwardToMulticast** function is associated with a location that contains video content and allows the MXK to receive video groups from the network. An interface with this value set to *true* should only transmit multicast traffic for which a JOIN request has been received. Any bridge interface with the **forwardToMulticast** parameter set to *false* discards multicast IP traffic. By default, the **forwardToMulticast** parameter is set to *true* on uplink bridges.

On the downlink bridge, the **learnMulticast** function is associated with interfaces that have hosts connected to them and allows the MXK to send video groups from downlink interfaces to the network. By default, the **learnMulticast** parameter is set to *true* on downlink bridges.

Note that JOIN operations enter on a **learnMulticast** interface associated with a downlink bridge and pass through on a **forwardToMulticast** interface associated with an uplink bridge.

The following table details various video bridge behaviors associated with different combinations of settings for the bridge parameters.

Table 8: learnMulticast-forwardToMulticast combinations and behavior

learnMulticast	forwardToMulticast	Behavior
False	False	The interface discards all incoming multicast packets and does not forward any of the packets.
True	False	The interface forwards both default multicast signaling packets and control multicast packets.
True	False	The interface discards incoming multicast content groups and forwards requested content groups.
False	True	The interface forwards control packets received on this interface to all other interfaces that have the learnMulticast field set to true .
False	True	The interface forwards content groups only to interfaces that have sent JOIN messages for a group.
True	True	Treat the same as an interface with the learnMulticast field set to false and the forwardToMulticast field set to true .

For the uplink bridge, note that the **forwardToMulticast** setting is *true* and the **learnMulticast** setting is *false*.

```
zSH> get bridge-interface-record ethernet2-800/bridge
bridge-interface-record ethernet2-800/bridge
vpi: -----> {0}
```

```

vci: -----> {0}
vlanId: -----> {800}
stripAndInsert: -----> {false}
customARP: -----> {true}
filterBroadcast: -----> {true}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {true}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {true}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

For the downlink bridge, the **forwardToMulticast** setting is *false* and the **learnMulticast** setting is *true*.

zSH> get bridge-interface-record 1-3-1-0-ads1-0-37/bridge

```

vpi: -----> {0}
vci: -----> {37}
vlanId: -----> {800}
stripAndInsert: -----> {true}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfConfigGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}

```

```
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {false}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
maxvideostreams: -----> {0}
mcasctrl: -----> {0}
....
```

In addition, you can run a **bridge igmp** command to determine whether IGMP is running on the system.

```
zSH> bridge igmp
```

VlanID	MAC Address	MCAST IP	Ifndx	Host MAC	Last Join
999	01:00:5e:02:7f:fe	224.2.127.254	921	00:02:02:0b:4a:a0	2
999	01:00:5e:02:7f:fe	224.2.127.254	922	00:02:02:0a:bb:6d	106
999	01:00:5e:02:7f:fe	224.2.127.254	923	00:02:02:0a:c0:b7	87
999	01:00:5e:02:7f:fe	224.2.127.254	924	00:02:02:0b:4e:c5	172
999	01:00:5e:02:7f:fe	224.2.127.254	925	00:02:02:0b:4c:7e	65
999	01:00:5e:02:7f:fe	224.2.127.254	926	00:02:02:0b:4f:08	46
999	01:00:5e:02:7f:fe	224.2.127.254	927	00:02:02:09:c1:7d	90
999	01:00:5e:02:7f:fe	224.2.127.254	928	00:02:02:0b:44:cd	71
999	01:00:5e:02:7f:fe	224.2.127.254	929	00:02:02:0b:4c:ca	61
999	01:00:5e:02:7f:fe	224.2.127.254	930	00:02:02:0b:47:bd	7
999	01:00:5e:02:7f:fe	224.2.127.254	931	00:02:02:0b:47:c7	177
999	01:00:5e:02:7f:fe	224.2.127.254	932	00:02:02:0b:4d:35	181
999	01:00:5e:02:7f:fe	224.2.127.254	933	00:02:02:0b:4d:5b	144
999	01:00:5e:02:7f:fe	224.2.127.254	934	00:02:02:0b:4a:a5	59
999	01:00:5e:02:7f:fe	224.2.127.254	935	00:02:02:0b:4c:9e	3
999	01:00:5e:02:7f:fe	224.2.127.254	936	00:02:02:09:c1:78	6
999	01:00:5e:02:7f:fe	224.2.127.254	937	00:02:02:0a:c0:ca	131

“Denial of Service” prevention

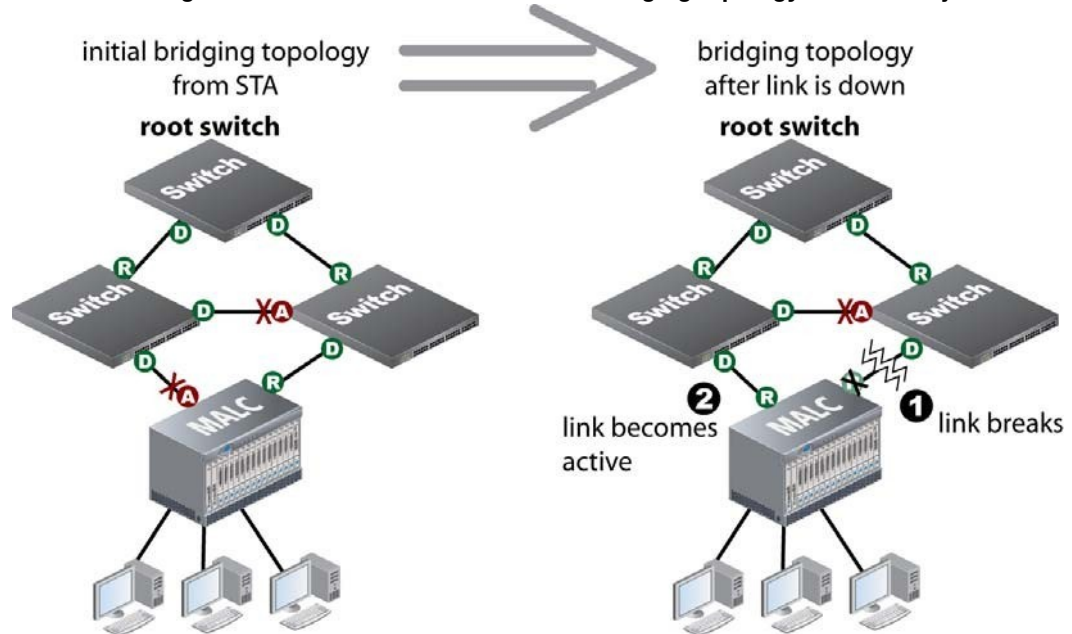
Enhanced broadcast storm protection the line cards prevents upstream broadcast storms. Broadcasts received into the system are placed in the lowest priority queue for exception packets. This queue is limited to 1,000 packets per second, the maximum number the hardware will allow onto the exception path. This throttling mitigates broadcast storms.

Rapid Spanning Tree Protocol (RSTP)

RSTP (802.1W) is an evolution of the Spanning Tree Protocol (STP, IEEE 802.1D). STP links network segments and eliminates one of the difficulties of configuring bridge topologies — bridge loops. There still can only be one active path. Once RSTP is configured for a bridged network the Spanning Tree Algorithm (STA) analyzes the network and determines which links should be active or not. The STA defines the links by configuring the ports.

In the bridged network the root bridge is selected. The STA sends out messages — Bridge Protocol Data Units (BPDU) — to determine the least cost path to the root bridge. From this analysis the port roles are determined.

Figure 28: The STA defines the initial bridging topology and later adjusts



RSTP port role

There are five port roles assigned by the STA to the port:

- ROOT: Root port

The root port is the closest to the root switch (also as root bridge. The root bridge is the only switch/bridge in the network that does not have a root port because it is the central bridge and root ports are defined by their relationship to the root bridge). The root port will receive the best BPDU from the root switch on a bridge.

In [Figure28](#), the root ports are designated with “R.”

For the STA to determine the root port for a device, five RSTP priority parameters are compared in the following priority sequence:

- 1) root bridge priority
- 2) root path cost
- 3) designated bridge priority
- 4) designated port ID
- 5) port priority

Only one RSTP port can be chosen as the root port per device. The port with the lowest value of RSTP priority parameters wins. If the first RSTP priority parameter have the same values on the ports, then the system will compare the next one, until it finds the root port.

- DSNT: Designated port

The designated port is the best port to send BPDU from the RSTP device to networked device.

In [Figure28](#), the designated ports are designated with “D.”

- ALT: Alternate port

The alternate port is a port that is blocked because it is receiving more useful BPDUs from another bridge. The alternate port can change to an active root port.

In [Figure28](#), the alternate ports are designated with “A” and are shown as blocked.

- BKP: Backup port

The backup port is a port that is blocked because it is receiving more useful BPDUs from the same bridge it is on. A backup port is only providing connectivity to the same network segment, so it cannot change to a root port.

- N/A: Not applicable

It means RSTP is not in the functional state yet. It usually will appear right after system bootup.

To view RSTP port roles, use **bridge show** command or **rstp-bridge show** command.

RSTP port state

IEEE 802.1w defines three port states in RSTP:

- DIS: RSTP discarding
- LRN: RSTP learning (a transitional state)
- FWD: RSTP forwarding (a normal operational state)

In operation there is no difference between a port with state DIS and one with state LRN as they both discard frames and do not learn MAC addresses. Ports which are blocking must keep transmitting BPDUs to retain maintain its port role and port state.

To show the RSTP port states, use **bridge show** command or **rstp-bridge show** command.

RSTP on uplinks

Rapid Spanning Tree Protocol (RSTP, IEEE 802.1W) is supported on upstream interface on the following MXK uplink cards:

- MXK-UPLINK-2X10G-8X1GE
- MXK-UPLINK-8X1GE
- MXK-UPLINK-4X1GE-CU
- MXK-UPLINK-4X1GE



Note: Interface 1-a-1-0/eth can not be used for RSTP. This interface is for inband management only.

Configuring RSTP on uplink bridges

The following example configures RSTP on uplink bridges.

- 1 Create RSTP uplink bridges on MXK upstream ports *1-a-4-0/eth* and *1-a-5-0/eth*:

Use **stp-bridge add interface/type uplink vlan x <tagged>** to add a VLAN interface to the upstream interface.

```
zSH> stp-bridge add 1-a-4-0/eth uplink vlan 500
Adding bridge on 1-a-4-0/eth
Created bridge-interface-record ethernet4-500/bridge
Bridge-path added successfully
```

```
zSH> stp-bridge add 1-a-5-0/eth uplink vlan 500
Adding bridge on 1-a-5-0/eth
Created bridge-interface-record ethernet5-500/bridge
Bridge-path added successfully
```

The bridge-path is automatically with the parameter *default*.

Even if the parameter *tagged* is not specified, the uplink bridge is considered a tagged bridge and the bridge will appear as tagged when using **bridge show**.

- 2 Show the bridges, enter:

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
upl	Tagged 500	ethernet4-500/bridge	BLK		
upl	Tagged 500	ethernet5-500/bridge	FWD S	VLAN 500	default STP: ROOT

Port *1-a-5-0* has been chosen as the root port, which is an active uplink port is receiving and forwarding packets. Port *1-a-4-0* is the alternate port, which is blocked and discarding packets.

- 3 To get detail RSTP information, use **stp-bridge show** command.

```
zSH> stp-bridge show
Bridge is running IEEE 802.1W RSTP
Bridge ID has priority 36000, address 00:01:47:1a:fe:64
Configured: hello=2, forward=15, max_age=20
Current root has priority 32768, address 00:02:16:95:b3:c0
```

```

Cost of root path 200019
1 bridge(s) present first-> ethernet4-500:
    Port is DOWN!
1 bridge(s) present first-> ethernet5-500:
    is a ROOT PORT in FORWARDING state
Root bridge has priority 32768, address 00:02:16:95:b3:c0
Designated bridge has priority 32768, address 00:03:e3:97:bb:00
Designated Port id is 32781:144, root path cost is 19
Timers: forward delay is 15, hello time is 2, message age is 1
sync: 0 synced: 0 reRoot: 0 rrWhile: 14 operEdge: 0 fdWhile: 0
learn: 1 forward: 1 agreed: 0 learning: 1 forwarding: 1 updtInfo: 0 selected: 1

```

Five RSTP priority parameters in these two ports will be compared in this sequence: Root bridge priority -> Root path cost -> Designated bridge priority -> Designated port ID -> Port priority.

In the above example, the value of the root bridge priority parameter is same on the two ports. Then, system compares the root path cost, since ethernet2-500 has the lower root path cost value 0, it becomes the root port.

- 4 If the first four RSTP priority parameters are the same, then the system compares the last parameter- port priority. The port with the lowest port priority wins. The port priority will be displayed when use **get stp-bind** *<profile-storage-key>* command, and can be changed use **update stp-bind** *<profile-storage-key>* command.

To verify the port priority in the **stp-bind** profile, enter:

```

zSH> get stp-bind ethernet4
stp-bind ethernet4/linegroup/0
portPriority: -> {128}

```

```

zSH> get stp-bind ethernet5
stp-bind ethernet5/linegroup/0
portPriority: -> {144}

```

To change the port priority in the **stp-bind** profile, enter:

```

zSH> update stp-bind ethernet4
stp-bind ethernet4/linegroup/0
Please provide the following: [q]uit.
portPriority: -> {128}: 160
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

- 5 To show the global RSTP parameters in the **stp-params** profile, use **get stp-params** *<profile-storage-key>* command.

```

zSH> get stp-params 0
stp-params 0
name: -----> {}
revision: -----> {0}
bridgePriority: -> {36000}

```

```

forceVersion: ---> {2}
fwdDelay: -----> {15}
helloTime: -----> {2}
migrateTime: ----> {3}
txHoldCount: ----> {3}
maxAge: -----> {20}

```

6 Delete the **stp-bridge(s)** on the ports.

```

zSH> stp-bridge delete ethernet4-500/bridge
Bridge-path deleted successfully
ethernet4-500/bridge delete complete

```

```

zSH> stp-bridge delete ethernet5-500/bridge
Bridge-path deleted successfully
ethernet5-500/bridge delete complete

```

RSTP rlinks

With the RSTP rlink in a ring configuration, instead of having a second redundant cloud link at each device, traffic can proceed through the other SLMS devices in the same network, which has its own uplink bridge.

See [Figure29](#) for an RSTP rlink ring topology. In this example, there is the mixed use of MALC and MXK in a network. Each MALC and MXK has a bridge interface with the characteristics of an uplink bridge enabled on the port, and an intralink bridge on another port. With RSTP rlink enabled on the intralink bridge, the intralink interface designated B2 on the MXK will be blocked, preventing looped bridge traffic. Traffic from the root switch arriving on MXK A1 would be checked for destination MAC match for local ports (downlinks) and if a match is not found, the packet would be dropped. Traffic from downstream bridges on MXK would be sent upstream towards the root switch out the interface B1. Traffic from downstream bridges on MALC would be sent upstream towards the root switch out the interface A1

Figure 29: RSTP rlink ring topology

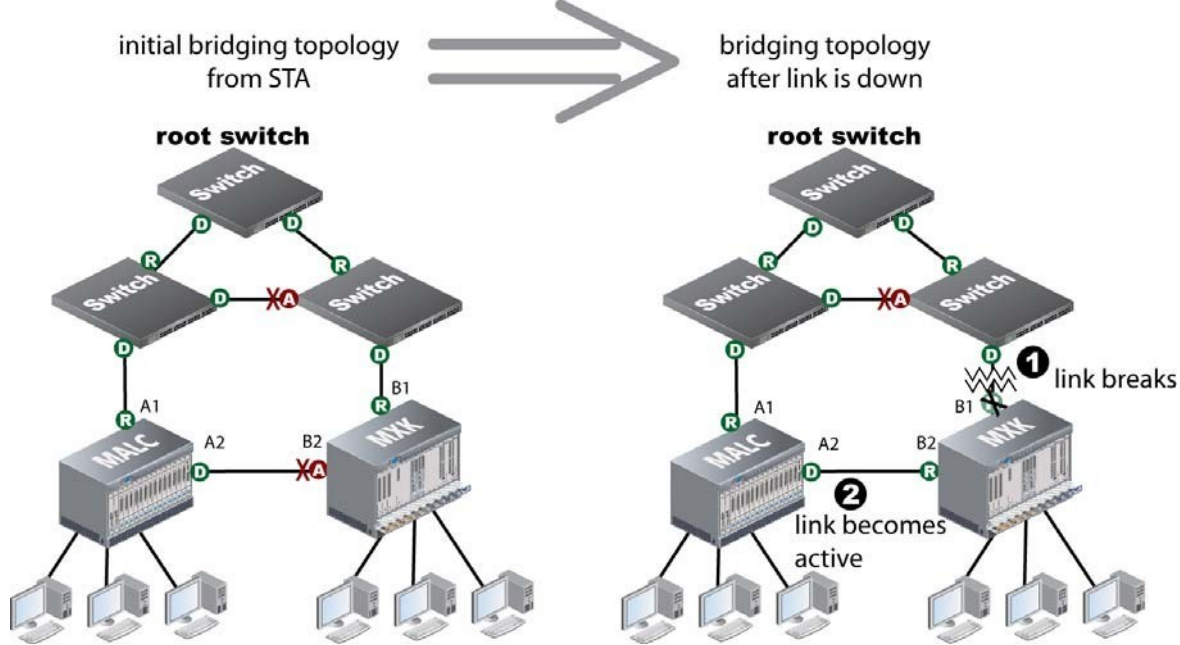
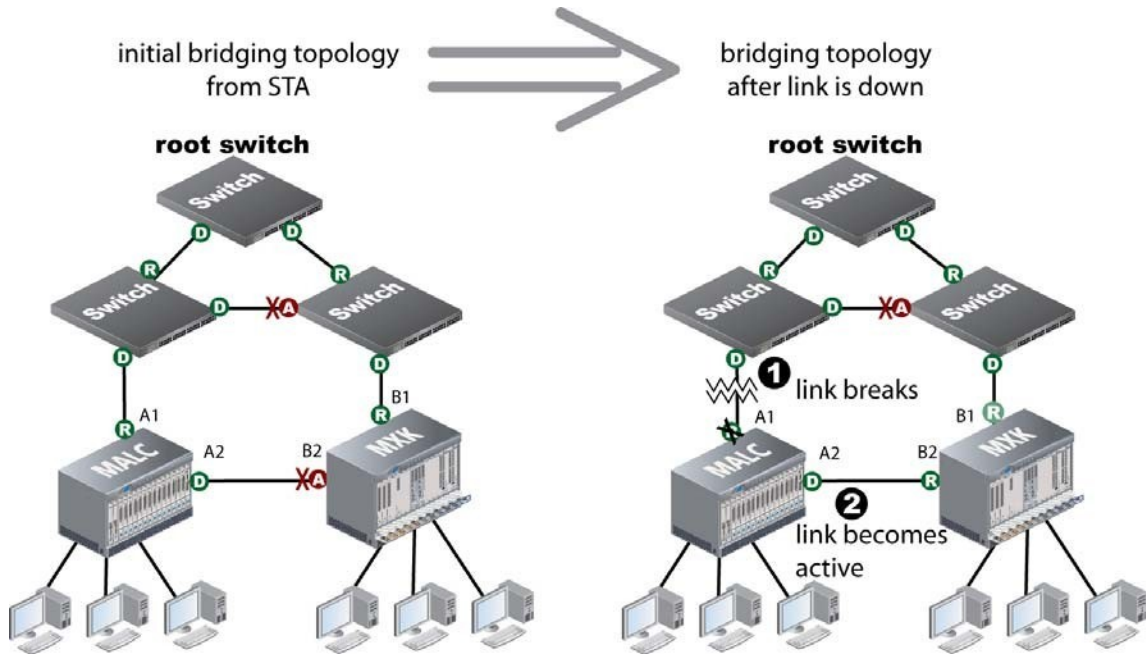


Figure29 also shows that if the connection from MXK to the root switch becomes unavailable, then the RSTP ring protocol will take the port B2 on the MXK out of the blocking state and into a forwarding state. Traffic from downlink bridges on MXK will no longer leave on B1. Instead, downstream traffic will be forwarded on B2 heading towards A2, and then sent upstream towards the root switch out the MAC’s root port interface A1.

Figure 30: RSTP rlink with a different downed link



Configuring RSTP rlinks

The configuration procedures for the RSTP rlink topologies are listed below.



Note: That this example show RSTP rlinks configured on both uplink and intralink ports on the MALC and MXK. You can also configure pure RSTP on the uplink port, and configure RSTP rlink on the intralink port.

- 1 As shown in [Figure29](#), on the MALC, to configure RSTP rlinks on uplink and intralink bridges, perform the following tasks:
 - a Create RSTP rlink on upstream port A1 (1-1-2-0) and intralink port A2 (1-1-3-0) with **stp-bridge add interface/type rlink vlan id <tagged>**.

```
zSH> stp-bridge add 1-1-2-0/eth rlink vlan 500
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record ethernet1-2-500/bridge
```

```
zSH> stp-bridge add 1-1-3-0/eth rlink vlan 500
Adding bridge on 1-1-3-0/eth
Created bridge-interface-record ethernet1-3-500/bridge
```

If the parameter *vlan id* is not specified, VLAN 0 is used. And if parameter *tagged* is not specified, the uplink bridge is considered a tagged bridge.

- b Create the bridge-paths for the rlink bridges using **bridge-path add interface/type global-rlink**.

```
zSH> bridge-path add ethernet1-3-500/bridge global-rlink
Bridge-path added successfully
Bridge-path added successfully
```

```
zSH> bridge-path add ethernet1-2-500/bridge global-rlink
Bridge-path added successfully
Bridge-path added successfully
```

- c View the bridge-paths.

```
zSH> bridge-path show
```

VLAN/SLAN	Bridge	Address
Global	ethernet1-3-500/bridge	Default
Global	ethernet1-3-500/bridge	Intralink
Global	ethernet1-2-500/bridge	Default
Global	ethernet1-2-500/bridge	Intralink

- d Show the baseline of the system, enter:

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
rlk	Tagged 500	ethernet2-500/bridge	FWD	S	Global default STP: ROOT

rlk Tagged 500 ethernet3-500/bridge DIS STP: ALT

Port A1 (1-1-2-0) has been chosen as the root port, which is an active uplink port in the forwarding state. Port A2 (1-1-3-0) is the intralink port and blocked by RSTP rlink topology to prevent loop. The state for this port is discarding. The role for this port is alternate.

- 2 On the MXK, to configure RSTP rlinks on uplink and intralink bridges, perform the following tasks:
- a To create RSTP rlink on upstream port B1(1-a-4-0) and intralink port B2 (1-a-5-0):

```
zSH> stp-bridge add 1-a-4-0/eth rlink vlan 500
Adding bridge on 1-a-4-0/eth
Created bridge-interface-record ethernet4-500/bridge
```

```
zSH> stp-bridge add 1-a-5-0/eth rlink vlan 500
Adding bridge on 1-a-5-0/eth
Created bridge-interface-record ethernet5-500/bridge
```

- b Add the following bridge-path(s):

```
zSH> bridge-path add ethernet4-500/bridge vlan 500 rlink
Bridge-path added successfully
Bridge-path added successfully
```

```
zSH> bridge-path add ethernet5-500/bridge vlan 500 rlink
Bridge-path added successfully
Bridge-path added successfully
```

- c Verify the bridge paths created, enter:

```
zSH> bridge-path show
VLAN/SLAN      Bridge      Address
-----
500 ethernet5-500/bridge      Default
500 ethernet5-500/bridge      Intralink
500 ethernet4-500/bridge      Default
500 ethernet4-500/bridge      Intralink
```

- d Show the baseline of the system, enter.

```
zSH> bridge show
Type VLAN      Bridge      St   Table Data
-----
rlk Tagged 500      ethernet5-500/bridge      FWD S VLAN 500 default STP: ROOT
rlk Tagged 500      ethernet4-500/bridge      FWD S VLAN 500 Intralink
STP: DSNT
```

Port B1 (1-a-5-0) has been chosen as the root port, which now is the closest port towards the root switch in terms of the root path cost. It can receive the best BPDUs from the root switch. Port B2 (1-a-4-0) is the intralink port has the designated port role, it can send and forward the best BPDUs.

- 3 As shown in [Figure30](#), if the connection between the MALC uplink port A1 to the root switch is broken, the intralink port A2 on the MALC will be blocked and start to forward traffic from downlink bridges to MXK intralink port B2, since the MXK is the closest device to the root switch now.
 - a On the MALC, verify uplink port A1(1-1-5-0) is down, intralink port A2 (1-1-4-0) is in the forwarding state and takes over the role of root port, enter.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
rlk Tagged	500	ethernet5-500/bridge	DWN		
rlk Tagged	500	ethernet4-500/bridge	FWD	S	Global default STP: ROOT

- b On the MXK, the port states and port roles will be same as before.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
rlk Tagged	500	ethernet5-500/bridge	FWD	S	VLAN 500 default STP: ROOT
rlk Tagged	500	ethernet4-500/bridge	FWD	S	VLAN 500 Intralink STP: DSNT

- 4 If you want to delete an RSTP rlink bridge, make sure to delete the uplink bridge path on bridge first, then delete the stp-bridge on the port.
 - a To delete the bridge path on MALC, use **bridge-path delete interface/bridge global-rlink** command.

```
zSH> bridge-path delete ethernet2-500/bridge link
Delete complete
Delete complete
```

To delete the bridge on MALC, use **stp-bridge delete interface/bridge** command.

```
zSH> stp-bridge delete ethernet2-500/bridge
ethernet2-500/bridge Delete complete
```

- b To delete the bridge path on MXK, use **bridge-path delete interface/bridge vlan x rlink** command.

```
zSH> bridge-path delete ethernet4-500/bridge vlan 500 rlink
Delete complete
Delete complete
```

To delete the bridge on MXK, use **stp-bridge delete interface/bridge** command.

```
zSH> stp-bridge delete ethernet4-500/bridge
ethernet4-500/bridge Delete complete
```

Bridging differences between the MALC and MXK

The MALC and the SLMS devices which have a similar architecture — the MALC XP, Raptor XP, and EtherXtend 34xx — have behaviors which are different than the MXK. The MXK processes one tag at a time. If double tags are present, the MXK processes based on the outer tag (*stag*) only. Because only the outer tag is processed, an Ethernet frame with an SLAN 200 and VLAN 75 only forwards the frame based on the SLAN 200.

MXK bridging configurations

This section includes the following bridging topics:

- [Configure a tagged uplink bridge with VLAN ID, page274](#)
- [Configure tagged or untagged downlink bridges with VLAN IDs, page276](#)
- [Configure tagged downlink bridges on Active Ethernet, page277](#)
- [Configure bridges using Q-in-Q \(VLAN IDs and SLAN IDs\), page282](#)
- [Configure bridges using VLAN 0, page290](#)
- [Configure TLS bridges, page300](#)
- [Configure TLS wire bridges, page301](#)
- [TLS bridge parameters floodUnknown and floodMulticast, page302](#)
- [Configure link aggregation bridges, page304](#)
- [Configure bridge loop issue prevention, page306](#)
- [Dynamic IP filtering on a bridge \(Secure DHCP\), page307](#)
- [Broadcast suppression, page309](#)

Configure a tagged uplink bridge with VLAN ID

All uplink bridges on the MXK require a VLAN ID. There must be an uplink bridge with a VLAN ID to match any existing downlink bridges with a VLAN ID in order to pass traffic. All uplink bridges default to *tagged* with the **stripAndInsert** parameter set to *false*. This means that the VLAN ID remains and is passed up to the network.

On the MXK, all bridge paths are set to *default*.



Note: It is recommended not to change bridge default settings unless advanced bridge configuration is required.

See [Bridge add and bridge-path add on page227](#) for when to accept the automatically created bridge path default configuration, and when it is

necessary to enter the **bridge-path add** command to create additional configurations.

Creating a uplink bridge

Create an uplink bridge and then create the bridge path for the uplink.

- 1 Create the uplink bridge, then verify the bridge.

```
zSH> bridge add 1-a-2-0/eth uplink vlan 55
Adding bridge on 1-a-2-0/eth
Created bridge-interface-record ethernet2-55/bridge
```

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
dwn	Tagged 55	1-1-2-0-eth-55/bridge	PND		
upl	Tagged 55	ethernet2-55/bridge	DWN		
dwn		55 1-1-1-0-eth/bridge	PND		

- 2 Add the bridge path on the uplink using *default* as the variable.

```
zSH> bridge-path add ethernet2-55/bridge vlan 55 default
Bridge-path added successfully
```

The *default* setting specifies this uplink receives all traffic with the designated VLAN ID from the downlinks.



Note: The MXK does not support the *global* variable.

- 3 Verify the **bridge-path** interface.

```
zSH> bridge-path show
```

VLAN/SLAN	Bridge	Address
55	ethernet2-55/bridge	Default

- 4 View the uplink bridge **bridge-interface-record** profile **stripAndInsert** parameter.

```
zSH> get bridge-interface-record 1-1-2-0-eth-55/bridge
bridge-interface-record 1-1-2-0-eth-55/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {55}
stripAndInsert: -----> {false}
```

Creating this uplink bridge with the **bridge add** command inserts 55 into the **vlanId** parameter and sets the **stripAndInsert** parameter to *false* in the **bridge-interface-record** profile.

Configure tagged or untagged downlink bridges with VLAN IDs

You can configure downlink bridges on the MXK using the variables *tagged* or *untagged* depending on the downstream configuration and the downstream bridging behavior that you desire. See [Configuring an Active Ethernet untagged downlink VLAN bridge on page 276](#) and [Configuring an Active Ethernet tagged downlink VLAN bridge on page 277](#) for configuration procedures.

Untagged downlink bridges on Active Ethernet

You configure downlink bridges as *untagged* when the downstream device does not expect or recognize VLAN IDs. Specifying a downlink bridge as *untagged* sets the **stripAndInsert** parameter in the **bridge-interface-record** to *true* causing the VLAN ID to be stripped out of the Ethernet packet on the way to the downstream device because it is not needed by the downstream device.

When a packet is sent back toward the upstream connection, that VLAN ID is inserted back into the Ethernet packet. If the correct VLAN ID is not on the packet traveling in the downstream direction, the packet will be dropped and not sent on to the downstream device. If that correct VLAN ID is not inserted back into the Ethernet packet traveling in the upstream direction, the uplink drops the packet.

The default for downlink bridges is *untagged*. Not designating either *untagged* or *tagged* when entering **bridge add interface/type downlink** always defaults to *untagged*. For example, both of these entries exhibit exactly the same bridging behavior.

```
zSH> bridge add 1-1-1-0/eth downlink vlan 55
```

and

```
zSH> bridge add 1-1-2-0/eth downlink vlan 55 untagged
```

In both cases the **stripAndInsert** parameter in the **bridge-interface-record** sets to *true*.

Entering **bridge add interface/type downlink** with the *tagged* variable sets the **stripAndInsert** parameter in the **bridge-interface-record** to *false*, causing the VLAN ID to remain in the Ethernet packet. Both the upstream and downstream devices will recognize and accept the Ethernet packet.

Configuring an Active Ethernet untagged downlink VLAN bridge

Untagged downlink bridges are usually configured on Active Ethernet. To configure an untagged downlink bridge with a VLAN ID:

- 1 To create an *untagged* bridge for downstream connections enter **bridge add interface/type downlink vlan <id>**.

```
zSH> bridge add 1-1-1-0/eth downlink vlan 55
Adding bridge on 1-1-1-0/eth
Created bridge-interface-record 1-1-1-0-eth/bridge
```

This example creates an untagged downlink interface on the Active Ethernet port 1 with a VLAN ID of 55.

- 2 To verify the downlink bridge, enter **bridge show**.

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
dwn          55 1-1-1-0-eth/bridge                PND
```

- 3 To view the **stripAndInsert** setting for the downlink bridge, enter.

```
zSH> get bridge-interface-record 1-1-1-0-eth/bridge
bridge-interface-record 1-1-1-0-eth/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {55}
stripAndInsert: -----> {true}
```

The **vlanId** parameter is set to 55, and the **stripAndInsert** parameter is set to *true*, meaning the VLAN ID 55 on this downstream bridge will be stripped on the downstream and inserted on the upstream.



Note: It is recommended not to change the default settings unless advanced bridge configuration is required.

Configure tagged downlink bridges on Active Ethernet

You configure a downlink bridge as *tagged* when a VLAN ID is expected or needed in the downstream configuration.

Designating a downlink bridge as *tagged*, sets the **stripAndInsert** parameter to *false*. This means that the VLAN ID is not stripped out of the Ethernet packet, and is delivered intact to a device expecting traffic with the designated VLAN ID. The VLAN ID remains unchanged when traveling in the upstream direction.

Configuring an Active Ethernet tagged downlink VLAN bridge

- 1 Create a *tagged* downlink bridge with a VLAN ID.

```
zSH> bridge add 1-1-2-0/eth downlink vlan 55 tagged
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record 1-1-2-0-eth-55/bridge
```

- 2 To display the tagged downlink bridge, enter **bridge show**.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
dwn	Tagged 55	1-1-2-0-eth-55/bridge	PND		
dwn	55	1-1-1-0-eth/bridge	PND		

3 View the **stripAndInsert** parameter of the **bridge-interface-record**.

```
zSH> get bridge-interface-record 1-1-2-0-eth-55/bridge
```

```
bridge-interface-record 1-1-2-0-eth-55/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {55}
stripAndInsert: -----> {false}
```

The VLAN ID parameter is set to 55. Since the downlink bridge is *tagged*, the **stripAndInsert** parameter is set to *false* and the VLAN ID is not stripped out of the Ethernet packet and remains intact in both directions.

Delete uplink and Active Ethernet downlink bridges

When you need to delete an uplink bridge, you must first delete the bridge path, then delete the uplink bridge.

Deleting an uplink bridge

1 Delete the uplink bridge-path before you delete the uplink bridge.

```
zSH> bridge-path delete ethernet2-55/bridge vlan 55 default
Delete complete
```

2 Delete the uplink bridge.

```
zSH> bridge delete ethernet2-55/bridge
ethernet2-55/bridge Delete complete
```

Deleting a downlink bridge

When you need to delete a downlink bridge, only the downlink bridge needs to be deleted.

Delete the downlink bridge.

```
zSH> bridge delete 1-1-1-0-eth/bridge
1-1-1-0-eth/bridge Delete complete
```

Configure uplink and downlink bridges on GPON for triple-play services

 **Note:** All bridges on GPON ports must have a VLAN ID and must be designated *tagged*. GPON does not support *untagged* bridging.

 **Note:** For information on Smart OMCI and ONU management, see [Chapter 9, MXK GPON Cards](#). For more information on configuring bridged video on the MXK, see [Chapter 6, Video Configuration](#).

You can create bridges on GEM ports to provide triple-play services. Bridges must be created to pass traffic between the MXK and the upstream data, voice, and video source, and the downstream ONUs.

You create the GEM port with **bridge add**. For different services, you can associate different GPON traffic profiles with different GEM ports.

 **Note:** If an ONU is activated with Smart OMCI, when you use **bridge add** to create a GEM port, be sure that the GEM port ID matches the GEM index specified in the Smart OMCI web-interface.

Configuring an uplink bridge and downlink bridge on a GEM port for data services

Create an uplink and downlink bridge on a GEM port for data services.

- 1 Create the tagged uplink bridge with a VLAN ID.

```
zSH> bridge add 1-a-4-0/eth uplink vlan 100 tagged
Adding bridge on 1-a-4-0/eth
Created bridge-interface-record ethernet4-100/bridge
```

- 2 Create the bridge path for the uplink bridge.

```
zSH> bridge-path add ethernet4-100/bridge vlan 100 default
Bridge-path added successfully
```

- 3 Create the GPON traffic profile for the downlink bridge for data services.

```
zSH> new gpon-traffic-profile 1
gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}: 512
traffic-class: -----> {ubr}:
compensated: -----> {false}:
shared: -----> {false}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

- 4 Create the downlink bridge with the GPON traffic profile and VLAN 100.

```
zSH> bridge add 1-9-1-501/gponport gtp 1 downlink vlan 100 tagged
GEM Port 1-9-1-501/gponport has been created on ONU 1-9-1-1/gpononu.
Adding bridge on 1-9-1-501/gponport
Created bridge-interface-record
1-9-1-501-gponport-100/bridge
```

Configuring an uplink bridge and downlink bridge on a GEM port for voice services

Create an uplink and downlink bridge on a GEM port for voice services.

- 1 Create the tagged uplink bridge with a VLAN ID.

```
zSH> bridge add 1-a-4-0/eth uplink vlan 200 tagged
Adding bridge on 1-a-4-0/eth
Created bridge-interface-record ethernet4-200/bridge
```

- 2 Create the default bridge path for the uplink bridge.

```
zSH> bridge-path add ethernet4-200/bridge vlan 200 default
Bridge-path added successfully
```

- 3 Create the GPON traffic profile for the downlink bridge for voice services.

```
zSH> new gpon-traffic-profile 2
gpon-traffic-profile 2
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}: 512
traffic-class: -----> {ubr}: cbr
compensated: -----> {false}: true
shared: -----> {false}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

- 4 Create the downlink bridge with the GPON traffic profile and VLAN 200.

```
zSH> bridge add 1-9-1-701/gponport gtp 2 downlink vlan 200
GEM Port 1-9-1-701/gponport has been created on ONU 1-9-1-1/gpononu.
Adding bridge on 1-9-1-701/gponport
Created bridge-interface-record 1-9-1-701-gponport/bridge
```

Configuring an uplink bridge and downlink bridge on a GEM port for video services

Create an uplink and downlink bridge on a GEM port for video services.

See [MXK bridged video on page 331](#) for complete details on creating bridged video.

- 1 Create the tagged uplink bridge with a VLAN ID.

```
zSH> bridge add 1-a-4-0/eth uplink vlan 300 tagged
Adding bridge on 1-a-4-0/eth
```


Created bridge-interface-record ethernet4-300/bridge

- 2 Create the bridge path for the uplink bridge and set the multicast aging period and IGMP query interval.

```
zSH> bridge-path addethernet4-300/bridge vlan 300 default igmptimer 30
igmpsnooping enable
Bridge-path added successfully
```

- 3 Create a multicast control list to define which multicast addresses the remote-end video can access.

```
zSH> new mcast-control-entry 1/1
mcast-control-entry 1/1
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.1
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

- 4 Create the GPON traffic profile for the downlink bridge for video services.

```
zSH> new gpon-traffic-profile 3
gpon-traffic-profile 3
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}: 512
traffic-class: -----> {ubr}: cbr
compensated: -----> {false}: true
shared: -----> {false}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

- 5 Create the downlink bridge with the GPON traffic profile and VLAN 300 and add the multicast control list and maximum video streams using the m/n format.

```
zSH> bridge add 1-9-1-901/gponport gtp 3 downlink vlan 300 video 1/6
GEM Port 1-9-1-901/gponport has been created on ONU 1-9-1-1/gpononu.
Adding bridge on 1-9-1-901/gponport
Created bridge-interface-record 1-9-1-901-gponport/bridge
```

Verify the configuration

Verify the configuration.

- 1 Verify the uplink and downlink bridges.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
dwn	Tagged 100	1-9-1-501-gponport-100/bridge	DWN		
dwn	200	1-9-1-701-gponport/bridge	DWN		

```
upl Tagged 100 ethernet4-100/bridge UP S VLAN 100 default
upl Tagged 200 ethernet4-200/bridge UP S VLAN 200 default
upl Tagged 300 ethernet4-300/bridge UP S VLAN 300 default
dwn          300 1-9-1-901-gponport/bridge DWN
```

2 Verify the GEM ports and their associated traffic profiles for the ONU.

```
zSH> gpononu gempports9/1/1
Slot 9 olt 1
```

ID	Onu	GEM Port	Admin	traf prof	Bandwidth Mbps/sec	traf class	compn	share	allocId
1	1-9-1-1	1-9-1-501	Up	1	0.5	ubr	False	False	501
		1-9-1-901	Up	3	0.5	cbr	True	False	901
		1-9-1-701	Up	2	0.5	cbr	True	False	701

Configure bridges using Q-in-Q (VLAN IDs and SLAN IDs)

The IEEE 802.1Q-in-Q VLAN tagging expands the VLAN space in the Ethernet frame to support the tagging of previously tagged packets. This second tag (SLAN) creates a "double-tagged" Ethernet frame.

In double-tagged or *staged* configurations, there is a VLAN ID and an SLAN ID. When the bridge interface with both a VLAN ID and an SLAN ID is configured to *tagged*, the **stripAndInsert** parameter for the VLAN ID is set to *false*, and the **s-tagstripAndInsert** parameter for the SLAN ID is set to *true*. In this case, the VLAN ID is *not* stripped and inserted and the SLAN ID is stripped and inserted. On the downlink this means that the VLAN ID is passed down, but the SLAN ID is not. The SLAN ID is stripped out for the egress traffic, and inserted back for the ingress traffic.

When the bridge interface with both a VLAN ID and an SLAN ID is configured to *staged*, the **stripAndInsert** parameter for the VLAN ID is set to *false*, and the **s-tagstripAndInsert** parameter for the SLAN ID is also set to *false*. In this case, neither the VLAN ID nor the SLAN ID are stripped and inserted. Both the VLAN ID and the SLAN ID are passed to the downstream device.

The MXK also supports setting CoS values in the Ethernet SLAN headers for bridged packets. This service enables you to assign a service level or class of service (CoS) to an Ethernet SLAN that is transported across a uplink, intralink, or downlinked s-tagged bridge. The configured CoS level specifies the packet priority and queueing methods used to transport the packet through the Ethernet network. The MXK sets and preserves the CoS settings to ensure these settings are passed to other Ethernet devices in the network for QoS processing. See [Shaping Traffic: Class of Service Queueing on page235](#).

Q-in-Q parameters

For Q-in-Q VLAN tagging, the **bridge-interface-record** profile supports the following parameters:

s-tagTPID: -----> {0x8100} typically set to 8100

```

s-tagId: -----> {50} SLAN ID
s-tagStripAndInsert: -----> {true} Specifies whether or not to strip and insert
s-tagOutgoingCOSOption: -----> {s-tagdisabled} Specifies whether to insert CoS value bits on
outgoing s-tag packets.
s-tagIdCOS: -----> {0} Specifies the CoS ID associated with the SLAN ID
s-tagOutgoingCOSValue: -----> {0} Specifies the value used to overwrite any existing CoS value in
outgoing s-tag packets

```

Turn off Q-in-Q for the entire MXK system

Setting the **options** parameter in the **system 0** profile to *cvlanonly* turns off the ability to configure bridges with SLAN IDs.

```

zSH> update system 0
system 0
Please provide the following: [q]uit.
syscontact: -----> {Zhone Global Services and Support 7001 Oakport
Street Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113
support@zhone.com}:
sysname: -----> {Zhone MxK}:
syslocation: -----> {Oakland}:
enableauthtraps: -----> {disabled}:
setserialno: -----> {0}:
zmsexists: -----> {false}:
zmsconnectionstatus: --> {inactive}:
zmsipaddress: -----> {0.0.0.0}:
configsyncexists: -----> {false}:
configsyncoverflow: ---> {false}:
configsyncpriority: ---> {high}:
configsyncaction: -----> {noaction}:
configsyncfilename: ---> {}:
configsyncstatus: -----> {syncinitializing}:
configsyncuser: -----> {}:
configsyncpasswd: -----> {** private **}: ** read-only **
numshelves: -----> {1}:
shelvesarray: -----> {}:
numcards: -----> {3}:
ipaddress: -----> {0.0.0.0}:
alternateipaddress: ---> {0.0.0.0}:
countryregion: -----> {us}:
primaryclocksource: ---> {0/0/0/0/0}:
ringsource: -----> {internalringsourcelabel}:
revertiveclocksource: -> {true}:
voicebandwidthcheck: --> {false}:
alarm-levels-enabled: -> {critical+major+minor+warning}:
userauthmode: -----> {local}:
radiusauthindex: -----> {0}:
secure: -----> {disabled}:
webinterface: -----> {enabled}:
options: -----> {NONE(0)}: cvlanonly
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Please reboot the system for cvlanonly change to take effect.
Record updated.

```

```

zSH> systemreboot
Do you want to reboot the system? (yes or no) [no]  yes
Do you want to exit from this request? (yes or no) [yes]  no
Are you sure? (yes or no) [no]  yes

```

If you now attempt to create a bridge with an SLAN ID, you will get the following error message:

```

zSH> bridge add 1-13-6-0/eth downlink vlan 777 slan 20
Adding bridge on 1-13-6-0/eth
Error: slan must be 0 for untag interface.

```

Q-in-Q bridging configurations on Active Ethernet

The MXK supports two ways of configuring Q-in-Q in bridging. The first way uses the *tagged* variable and the second way uses the *staged* variable. Some MXK bridging configurations are from an *staged* bridge to a *tagged* bridge (see [Tagged downlink to staged uplink bridge configuration \(tagged to staged\) on page284](#)), or from a *staged* bridge to a *staged* bridge (see [staged downlink bridge and staged uplink bridge \(stag to stag\) on page287](#)).

These bridge types can go from uplink to downlink or from downlink to uplink depending on your configuration requirements.

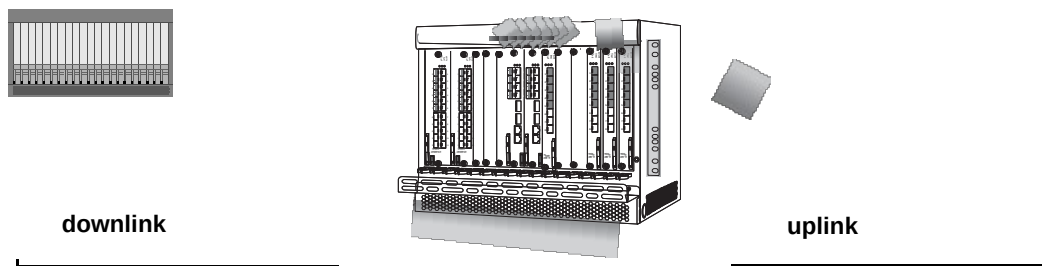
Tagged downlink to staged uplink bridge configuration (tagged to staged)

[Figure31](#) shows an example of using Q-in-Q (SLAN IDs) on both the uplink and the downlink bridge, but designating *tagged* on the downlink bridge and *staged* on the uplink bridge.

In this case, designating the downlink bridge as *tagged* sets the SLAN ID **s-tagstripAndInsert** parameter to *true* and the VLAN ID **stripAndInsert** parameter to *false* in the **bridge-interface-record** profile. This causes the SLAN ID to be stripped as it passes to the downstream device, and re-inserted when traveling in the upstream direction. The VLAN ID remains in both directions.

This type of configuration allows a downstream device such as a MALC to receive the VLAN ID and not the SLAN ID. [Figure31](#) shows a tagged downlink and staged uplink bridging configuration.

Figure 31: Tagged downlink and staged uplink example



```
bridge add 1-1-1-0/eth downlink vlan 101
slan 503 tagged
```

```
bridge add 1-a-2-0/eth uplink vlan 101
slan 503 staged
```

Configuring a tagged downlink and staged uplink bridge

This configuration will create a downlink bridge that strips out the SLAN ID on the downlink and re-inserts the SLAN ID when traveling to the uplink.

- 1 Create a *tagged* downlink bridge with an SLAN ID 501 and a VLAN ID 101 on the Active Ethernet card:

```
zSH> bridge add 1-1-1-0/eth downlink vlan 101 slan 501 tagged
Adding bridge on 1-1-1-0/eth
Created bridge-interface-record 1-1-1-0-eth-101/bridge
```

Designating the downlink bridge as *tagged* strips the SLAN ID on the way to the device and re-inserts the SLAN ID on the way to the uplink. The VLAN ID remains in both directions. The **stripAndInsert** parameter for the VLAN ID is *false*, and the **s-tagStripAndInsert** parameter for the SLAN ID is *true* in the **bridge-interface-record**:

```
zSH> get bridge-interface-record 1-1-1-0-eth-101/bridge
bridge-interface-record 1-1-1-0-eth-101/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {101}
stripAndInsert: -----> {false} strip and insert behavior
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
```

```
s-tagId: -----> {501}
s-tagStripAndInsert: -----> {true} stripped on the egress and inserted on the ingress
...
```

2 To verify the bridge enter:

```
zSH> bridge show
Type VLAN      Bridge
-----
dwn Tg  101/501 1-1-1-0-eth-101/bridge PND
```

3 Create an *staged* uplink bridge with a VLAN ID and a SLAN ID to match the downlink bridge:

```
zSH> bridge add 1-a-3-0/eth uplink vlan 101 slan 501 staged
Adding bridge on 1-a-3-0/eth
Created bridge-interface-record ethernet3-101-501/bridge
```

Designating the uplink bridge as *staged* does not strip or insert the either the VLAN ID or the SLAN ID. The **stripAndInsert** parameter for the VLAN ID is *false*, and the **s-tagStripAndInsert** parameter for the SLAN ID is also *false* in the **bridge-interface-record**:

```
zSH> get bridge-interface-record ethernet3-101-501/bridge
bridge-interface-record ethernet3-101-501/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {101}
stripAndInsert: -----> {false} strip and insert behavior
customARP: -----> {true}
filterBroadcast: -----> {true}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {true}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {true}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {501}
s-tagStripAndInsert: -----> {false} strip and insert behavior
...
```

4 Verify the bridge:

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
dwn Tg	101/501	1-1-1-0-eth-101/bridge	PND		
upl ST	101/501	ethernet3-101-501/bridge	DWN		

5 Create the bridge-path:

```
zSH> bridge-path addethernet3-101-501/bridge vlan 101 slan 501 default
Bridge-path added successfully
```

6 Verify the bridge-path:

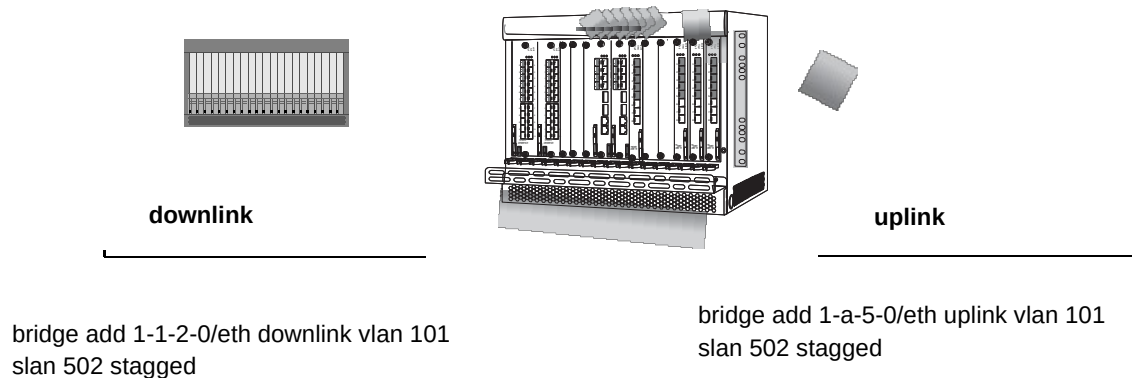
```
zSH> bridge-path show
```

VLAN/SLAN	Bridge	Address
101/501	ethernet3-101-501/bridge	Default

staged downlink bridge and staged uplink bridge (stag to stag)

[Figure32](#) describes an staged downlink to staged uplink bridging configuration.

Figure 32: staged to staged uplink downlink configuration



Configuring an staged bridge on the downlink and an staged bridge on the uplink

1 Create an *staged* downlink bridge with an SLAN ID and a VLAN ID:

```
zSH> bridge add1-13-1-0/eth downlink vlan 102 slan 502 staged
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth-102-502/bridge
```

Designating the downlink bridge as *staged* does not strip or insert the either the VLAN ID or the SLAN ID. The **stripAndInsert** parameter for the VLAN ID is *false*, and the **s-tagStripAndInsert** parameter for the SLAN ID is also *false* in the **bridge-interface-record**:

```
zSH> get bridge-interface-record-13-1-0-eth-102-502/bridge
bridge-interface-record 1-13-1-0-eth-102-502/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {102}
stripAndInsert: -----> {false} no strip and insert behavior
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {502}
s-tagStripAndInsert: -----> {false} no strip and insert behavior
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}
```

2 Verify the bridge:

```
zSH> bridge show
Type VLAN      Bridge
-----
tls Tagged 200  ipobridge-200/bridge      UP  D 00:01:47:17:ee:54
dwn ST  102/502  1-13-1-0-eth-102-502/bridge DWN
```

3 Create an *staged* uplink bridge with the same VLAN ID and SLAN ID as the downlink bridge:

```
zSH> bridge add1-a-5-0/eth uplink vlan 102 slan 502 staged
Adding bridge on 1-a-5-0/eth
Created bridge-interface-record ethernet5-102-502/bridge
```

Designating *staged* sets the **stripAndInsert** parameter for the VLAN ID and the **s-tagStripAndInsert** parameter for the SLAN ID to *false* in the **bridge-interface-record**:


```

zSH> get bridge-interface-record ethernet5-102-502/bridge
bridge-interface-record ethernet5-102-502/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {102} VLAN ID
stripAndInsert: -----> {false} strip and insert behavior
customARP: -----> {true}
filterBroadcast: -----> {true}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {true}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {true}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {502} VLAN ID
s-tagStripAndInsert: -----> {false} strip and insert behavior
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

4 Create the bridge path for the uplink bridge:

```

zSH> bridge-path add ethernet5-101-501/bridge vlan 101 slan 501 default
Bridge-path added successfully

```

5 Verify the bridge path:

```

zSH> bridge-path show

```

VLAN/SLAN	Bridge	Address
102/502	ethernet5-102-502/bridge	Default

Delete the uplink and downlink bridges

Deleting an uplink bridge

Delete the bridge-path then delete the uplink bridge.

- 1 View the existing bridge paths:

```
zSH> bridge-path show
VLAN/SLAN      Bridge                                     Address
-----
102/502        ethernet5-102-502/bridge                 Default
```

- 2 Delete the bridge path:

```
zSH> bridge-path delete ethernet5-102-502/bridge vlan 102 slan 502 default
Delete complete
```

- 3 Delete the bridge:

```
zSH> bridge delete ethernet3-101-501/bridge
ethernet3-101-501/bridge Delete complete
```

Configure bridges using VLAN 0

On the MXK, VLAN 0 functions as a wildcard that will recognize all VLAN IDs but can only be used in conjunction with an SLAN ID. You can designate VLAN 0 on uplink, downlink, TLS, and intralink bridges. Any bridge configuration using VLAN 0 can be designated either *tagged* or *staged* depending on the bridging behavior desired.



Note: VLAN 0 (VLAN wild card) is not supported on GPON.

MXK bridging configuration with VLAN 0 on uplink and downlink bridges

In situations where a business subscriber uses many internal VLAN IDs that the service provider does not care about, you would configure the downlink bridge with VLAN ID 0 and an SLAN ID. The SLAN ID will be recognized going out to the network and the VLAN IDs will be passed down to the business subscriber. In this scenario, designating the downlink bridge as *tagged* strips out SLAN ID to the business and reinserts the SLAN toward the network. The network will look for the outer tag, the SLAN ID and not care about the VLAN IDs.

Creating a tagged downlink bridge with VLAN 0 and an s-tagged uplink bridge with VLAN 0

- 1 Create the *tagged* downlink bridge with VLAN 0 and specify the SLAN ID.

```
zSH> bridge add 1-13-1-0/eth downlink vlan 0 slan 501 tagged
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth-0/bridge
```

Verify the bridge.

```
zSH> bridge show
Type VLAN          Bridge                               St  Table Data
-----
dwn Tg      0/501  1-13-1-0-eth-0/bridge                DWN
```

Verify the bridging behavior.

```
zSH> get bridge-interface-record-13-1-0-eth-0/bridge
bridge-interface-record 1-13-1-0-eth-0/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {0} VLAN 0
stripAndInsert: -----> {false} all vlans are passed to the business subscriber
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {501} SLAN ID
s-tagStripAndInsert: -----> {true} stripped out to the subscriber inserted to the network
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}
```

2 Create the staged uplink with VLAN 0 and SLAN ID.

```
zSH> bridge add 1-a-2-0/eth uplink vlan 0 slan 501 staged
Adding bridge on 1-a-2-0/eth
Created bridge-interface-record ethernet2-0-501/bridge
```

3 Create the default bridge path for the uplink bridge.

```
zSH> bridge-path add ethernet2-0-501/bridge vlan 0 slan 501 default
Bridge-path added successfully
```

Verify the bridge path.

zSH> **bridge-path show**

VLAN/SLAN	Bridge	Address
0/501	ethernet2-0-501/bridge	Default

Verify the uplink bridge.

zSH> **bridge show**

Type	VLAN	Bridge	St	Table	Data
upl ST	0/501	ethernet2-0-501/bridge	DWN	S	VLAN 0 default
dwn Tg	0/501	1-13-1-0-eth-0/bridge	DWN		

Verify the bridging behavior.

zSH> **get bridge-interface-record ethernet2-0-501/bridge**

```
bridge-interface-record ethernet2-0-501/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {0} VLAN ID 0
stripAndInsert: -----> {false} is passed to the network
customARP: -----> {true}
filterBroadcast: -----> {true}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {true}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {true}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {501} VLAN ID 501
s-tagStripAndInsert: -----> {false} is passed to the network
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}
```

Deleting the uplink downlink bridges with VLAN 0

If necessary, delete the uplink and downlink bridges.

- 1 Delete the bridge path.

```
zSH> bridge-path delete ethernet2-0-501/bridge vlan 0 slan 501 default
Delete complete
```

- 2 Delete the uplink bridge.

```
zSH> bridge delete ethernet2-0-501/bridge
ethernet2-0-501/bridge Delete complete
```

- 3 Delete the downlink bridge.

```
zSH> bridge delete 1-13-1-0-eth-0/bridge
1-13-1-0-eth-0/bridge Delete complete
```

MXK bridging configuration with VLAN 0 on TLS bridges for multi-point connections

In bridging configurations where multi-point connections are needed, you can configure TLS bridges with VLAN 0 and the same SLAN ID. A multi-point connection is two or more connections for the same SLAN ID facing the subscriber. The TLS bridge facing the subscriber is *tagged*. This means the SLAN ID is stripped out to the subscriber and inserted to the network. The TLS bridge to the network is *staged*, keeping both the VLANs and the SLAN ID. The network device will recognize the SLAN ID, i.e. the outer tag.

Creating TLS bridges for a multi-point connection

First create the TLS bridge with VLAN 0 and the SLAN ID on the network facing Ethernet port, then create the TLS bridges on the subscriber Active Ethernet ports with the same SLAN ID.

- 1 Create the *staged* TLS bridge on an Ethernet port facing the network.

```
zSH> bridge add 1-a-3-0/eth tls vlan 0 slan 200 staged
Adding bridge on 1-a-3-0/eth
Created bridge-interface-record ethernet3-0-200/bridge
```

Verify the bridge.

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
tls ST      0/200 ethernet3-0-200/bridge             DWN
```

Verify the **bridge-interface-record**.

```
zSH> get bridge-interface-record ethernet3-0-200/bridge
bridge-interface-record ethernet3-0-200/bridge
vpi: -----> {0}
vci: -----> {0}
```

```

vlanId: -----> {0}
stripAndInsert: -----> {false}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {false}
learnUnicast: -----> {true}
maxUnicast: -----> {100}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {200}
s-tagStripAndInsert: -----> {false}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {true}
floodMulticast: -----> {true}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

2 Create the *tagged* TLS bridges facing the subscriber.

```

zSH> bridge add1-13-1-0/eth tls vlan 0 slan 200 tagged
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth-0/bridge

```

```

zSH> bridge add1-13-2-0/eth tls vlan 0 slan 200 tagged
Adding bridge on 1-13-2-0/eth
Created bridge-interface-record 1-13-2-0-eth-0/bridge

```

```

zSH> bridge add1-13-3-0/eth tls vlan 0 slan 200 tagged
Adding bridge on 1-13-3-0/eth
Created bridge-interface-record 1-13-3-0-eth-0/bridge

```

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
tls Tg	0/200	1-13-1-0-eth-0/bridge	DWN		
tls Tg	0/200	1-13-3-0-eth-0/bridge	DWN		
tls ST	0/200	ethernet3-0-200/bridge	DWN		
tls Tg	0/200	1-13-2-0-eth-0/bridge	DWN		

Verify the **bridge-interface-record**.

```
zSH> get bridge-interface-record-13-3-0-eth-0/bridge
bridge-interface-record 1-13-3-0-eth-0/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {0}
stripAndInsert: -----> {false}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {false}
learnUnicast: -----> {true}
maxUnicast: -----> {100}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {200}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {true}
floodMulticast: -----> {true}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}
```

Deleting the TLS bridges

Delete the TLS bridges if necessary.

- 1 Delete the TLS bridge facing the network.

```
zSH> bridge delete ethernet3-0-200/bridge
ethernet3-0-200/bridge Delete complete
```

- 2 Delete the TLS bridges facing the subscriber.

```
zSH> bridge delete 1-13-1-0-eth-0/bridge
1-13-1-0-eth-0/bridge Delete complete
```

```
zSH> bridge delete 1-13-2-0-eth-0/bridge
1-13-2-0-eth-0/bridge Delete complete
```

```
zSH> bridge delete 1-13-3-0-eth-0/bridge
1-13-3-0-eth-0/bridge Delete complete
```

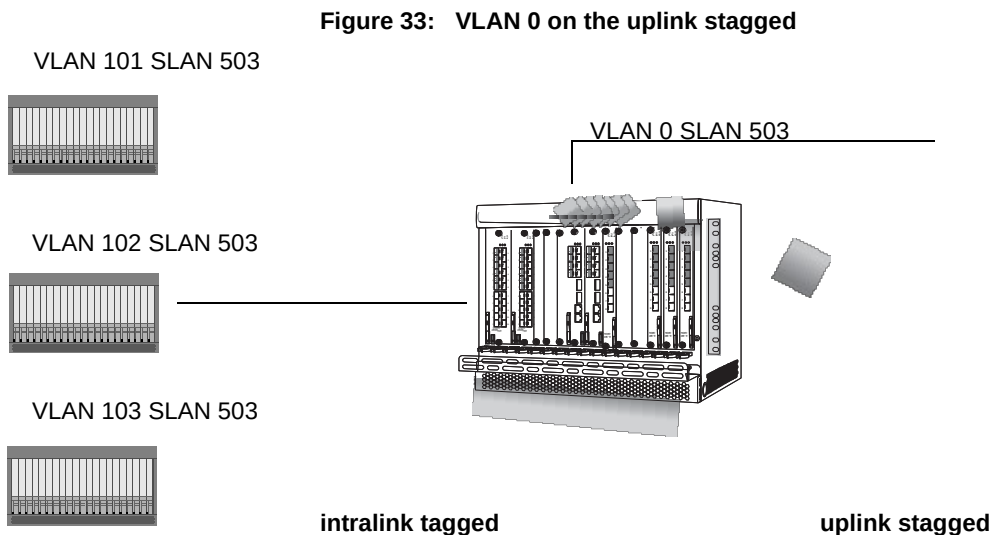
MXK bridging configuration with VLAN 0 on tagged intralinks

The MXK uses a VLAN wildcard, VLAN ID 0, on double-tagged (*staged*) uplink bridges. This is useful for creating several downstream bridges that have different VLAN IDs but have the same SLAN ID.



Note: Single-tagged VLAN 0 is not allowed.

For example, you might want to subtend several MALCs off of an MXK with different VLAN IDs but the same SLAN ID. In this case, VLAN ID 0 on the uplink bridge will accept all of the VLAN IDs and the same SLAN ID for each subtended MALC. This allows you to configure one uplink bridge that will recognize each of the VLAN IDs and the SLAN ID as shown in [Figure 33](#).



Configuring intralink bridges (tagged to staged configuration)

Creating *tagged* intralink bridges sets the **stripAndInsert** parameter to *false* for the VLAN ID and the **s-tagStripAndInsert** parameter for the SLAN ID to *true*. This causes the strip and insert behavior to strip the SLAN ID on the way to the subtended device and re-insert the SLAN ID on the way to the uplink. The VLAN ID is passed in both directions. The uplink bridge is *staged*, which sets the **stripAndInsert** parameter and the **s-tagStripAndInsert** parameter to *false*. Both the SLAN ID and the VLAN ID are passed on upstream.

- 1 Create *tagged* intralink bridges to the subtended devices:


```
zSH> bridge add1-1-5-0/eth intralink vlan 101 slan 503 tagged
Adding bridge on 1-1-5-0/eth
Created bridge-interface-record 1-1-5-0-eth-101/bridge
```

```
zSH> bridge add1-1-6-0/eth intralink vlan 102 slan 503 tagged
Adding bridge on 1-1-6-0/eth
Created bridge-interface-record 1-1-6-0-eth-102/bridge
```

```
zSH> bridge add1-1-7-0/eth intralink vlan 103 slan 503 tagged
Adding bridge on 1-1-7-0/eth
Created bridge-interface-record 1-1-7-0-eth-103/bridge
```

2 Verify the bridges:

```
zSH> bridge show
Type VLAN      Bridge                                     St  Table Data
-----
int Tg  101/503  1-1-5-0-eth-101/bridge                    PND
int Tg  102/503  1-1-6-0-eth-102/bridge                    PND
int Tg  103/503  1-1-7-0-eth-103/bridge                    PND
```

3 Create the bridge-path for each of the intralink bridges:

```
zSH> bridge-path add1-1-5-0-eth-101/bridge intralink
Bridge-path added successfully
```

```
zSH> bridge-path add1-1-6-0-eth-102/bridge intralink
Bridge-path added successfully
```

```
zSH> bridge-path add1-1-7-0-eth-103/bridge intralink
Bridge-path added successfully
```

4 Verify the bridge-path:

```
zSH> bridge-path show
VLAN/SLAN      Bridge                                     Address
-----
0 1-1-5-0-eth-101/bridge                    Intralink
0 1-1-6-0-eth-102/bridge                    Intralink
0 1-1-7-0-eth-103/bridge                    Intralink
```

5 Create the *staged* uplink bridge using VLAN ID 0:

```
zSH> bridge add1-a-7-0/eth uplink vlan 0 slan 503 staged
Adding bridge on 1-a-7-0/eth
Created bridge-interface-record ethernet7-0-503/bridge
```

The uplink bridge accepts any VLAN IDs with the same SLAN ID 503.

Delete bridge-path and bridge

To delete intralink bridges if necessary, first delete the intralink bridge-path, then delete the intralink bridge.

1 Delete the bridge-path, then delete the intralink bridge:

```
zSH> bridge-path delete1-1-5-0-eth-101/bridge intralink
Delete complete
```

```
zSH> bridge deleteethernet5-101-501/bridge
ethernet5-101-501/bridge Delete complete
```

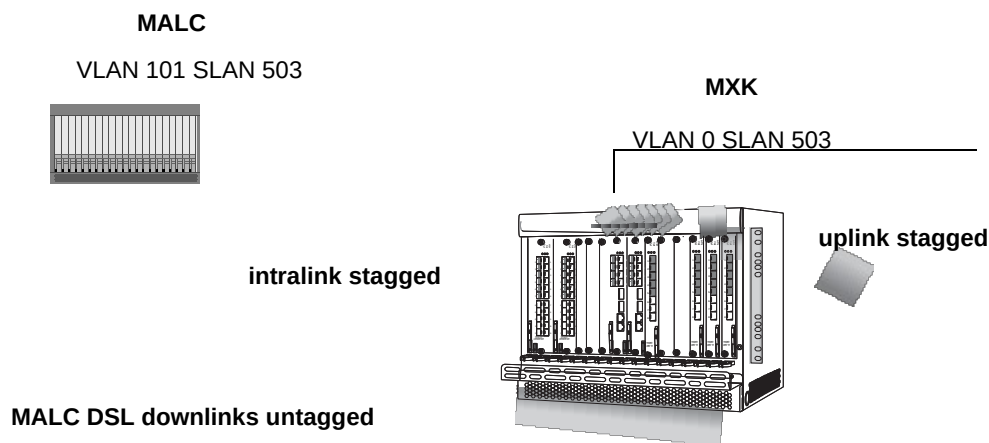
2 Delete the uplink bridge:

```
zSH> bridge deleteethernet7-0-503/bridge
ethernet7-0-503/bridge Delete complete
```

MXK bridging configuration with VLAN 0 on staged intralinks

In special cases, you can create *staged* intralink bridges from the MXK to subtended MALCs. You do this when there are *untagged* downlink bridges on the MALC to the downstream device, for example, on DSL lines to subscriber phones. In this case, the downstream devices on the MALC do not need the VLAN ID or SLAN ID, but are connected to a network that expects both an SLAN ID and a VLAN ID on the uplink as shown in [Figure 34](#).

Figure 34: Subtended MALCs off the MXK with staged intralinks



Creating the downlink bridge with a VLAN ID and an SLAN ID and using the variable *untagged* causes certain strip and insert behavior. For the *untagged* downlink bridge, both the **stripAndInsert** parameter and the **s-tagstripAndInsert** parameter are set to *true* which causes the VLAN ID and the SLAN ID to be stripped out in the downstream direction, and re-inserted in the upstream direction. Creating an intralink bridge using the variable *staged*, causes both the **stripAndInsert** parameter and the **s-tagstripAndInsert** parameter to be set to false, and both the SLAN ID and the VLAN ID are passed both downstream (to the MALC) and upstream (to the network). This strip and insert behavior on the downlink is called double promotion.



Note: Double promotion, or *untagged* bridges in a network using VLANs and SLANs can only occur on the MALC.

Configuring staged intralink bridge and staged uplink bridge on the MXK and untagged downlink bridge on the MALC

- 1 Create an *staged* uplink bridge and the bridge path for the uplink bridge on the MXK:

```
zSH> bridge add 1-a-4-0/eth uplink vlan 0 slan 503 staged
Adding bridge on 1-a-4-0/eth
Created bridge-interface-record ethernet4-0-503/bridge
```

```
zSH> get bridge-interface-record ethernet4-0-503/bridge
bridge-interface-record ethernet4-0-503/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {0}
stripAndInsert: -----> {false}
customARP: -----> {true}
filterBroadcast: -----> {true}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {true}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {true}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {503}
s-tagStripAndInsert: -----> {false}
...
```

- 2 Create an *staged* intralink bridge on the MXK:

```
zSH> bridge add 1-1-4-0/eth intralink vlan 101 slan 503 staged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-101-503/bridge
```

```
zSH> get bridge-interface-record 1-1-4-0-eth-101-503/bridge
bridge-interface-record 1-1-4-0-eth-101-503/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {101}
stripAndInsert: -----> {false}
customARP: -----> {false}
```

```

filterBroadcast: -----> {false}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {503}
s-tagStripAndInsert: -----> {false}
...

```

3 Create an *untagged* downlink bridge on the MALC:

```

zSH> bridge add1-9-1-0/eth downlink vlan 100 slan 500 untagged
Adding bridge on 1-9-1-0/eth
Created bridge-interface-record 1-9-1-0-eth/bridge

```

```

zSH> get bridge-interface-record 1-9-1-0-eth/bridge
bridge-interface-record 1-9-1-0-eth/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {100}
stripAndInsert: -----> {true}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {500}
s-tagStripAndInsert: -----> {true}
...

```

Configure TLS bridges

TLS bridges learn MAC addresses and forward packets to learned destinations. Broadcasts and unknown unicasts are flooded out all interfaces

except the ingress interface. Packets entering the system on a TLS interface have their source MAC addresses learned and associated with the interface so that frames from the network that come in on other TLS bridges in the VLAN can be sent to the correct interface.

TLS is a symmetrical bridge and can only be used with other TLS bridges.

Creating a TLS bridge configuration

- 1 Create a TLS bridge on the MXK Active Ethernet card.

```
zSH> bridge add 1-13-6-0/eth tls vlan 900
Adding bridge on 1-13-6-0/eth
Created bridge-interface-record 1-13-6-0-eth/bridge
```

- 2 Create a TLS bridge on the uplink card.

```
zSH> bridge add 1-13-6-0/eth tls vlan 900
Adding bridge on 1-13-6-0/eth
Created bridge-interface-record 1-13-6-0-eth/bridge
```

Deleting the TLS bridge configuration

- 1 Delete the bridge on the Active Ethernet card.

```
zSH> bridge delete 1-13-6-0-eth/bridge vlan 900
1-13-6-0-eth/bridge Delete complete
```

- 2 Delete the bridge on the uplink card.

```
zSH> bridge delete 1-13-6-0-eth/bridge vlan 900
1-13-6-0-eth/bridge Delete complete
```

Configure TLS wire bridges

A wire bridge is a reserved TLS bridge. When configuring wire bridges, the VLAN ID used on the two wire bridge interfaces is reserved for the entire device and cannot be used again. Wire bridges are confined to two bridge interfaces on a VLAN ID. Additional bridge interfaces on the VLAN ID cannot be added.

Configuring wire bridges

- 1 Create the first wire bridge interface with VLAN ID.

```
zSH> bridge add 1-a-5-0/eth wire vlan 500
Adding bridge on 1-a-5-0/eth
Created bridge-interface-record ethernet5/bridge
```

- 2 Create the second wire bridge interface with the same VLAN ID.

```
zSH> bridge add 1-13-1-0/eth wire vlan 500
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth/bridge
```

3 View the wire bridges.

```

zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
upl Tagged 800  ethernet2-800/bridge                DWN S VLAN 800 default
wre          500 ethernet5/bridge                UP  D 00:03:e3:97:bb:01
wre          500 1-13-1-0-eth/bridge                DWN

```

If a VLAN ID is used for two wire bridges, the system prevents that VLAN ID from being used again.

```

zSH> bridge add 1-13-2-0/eth wire vlan 500
Error: existing wire bridges on s/vlan 0/500 (2) and 0/ANY_VLAN wildcard (0)
exceeds 1.

```

TLS bridge parameters **floodUnknown** and **floodMulticast**

TLS bridges can provide VPN-like services with the **floodUnknown** and **floodMulticast** parameters that allow the MXK to forward unknown traffic to all bridge interfaces within the VLAN.

floodUnknown parameter

The **floodUnknown** parameter provides the ability to flood unknown unicast destination frames with unknown unicast MAC addresses to all interfaces on the VLAN. One case where this may need to be done is when voice packets are flooded out on the VLAN to see if there is an interface that will respond.

When the **floodUnknown** parameter is set to *true*, the MXK forwards (floods) frames with unknown unicast MAC addresses to all interfaces on the VLAN. The **learnUnicast** parameter is set to *true*. If a interface responds to a flooded packet, the address is learned, and that packet does not need to be flooded again.

When this parameter is set to *false*, the MXK discards frames with an unknown unicast MAC addresses. Frames that do not find a match in the forwarding table are discarded.

For TLS bridges, the default setting for these parameters is *true*. For uplink downlink, and intralink bridges, the default setting for these parameters is *false*.

This example shows that the **floodUnknown** and **learnUnicast** default settings are set to true on a TLS bridge.

```

zSH> bridge add 1-13-1-0/eth tls vlan 500
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth/bridge

zSH> get bridge-interface-record-13-1-0-eth/bridge
bridge-interface-record 1-13-1-0-eth/bridge
vpi: -----> {0}
vci: -----> {0}

```

```

vlanId: -----> {500}
stripAndInsert: -----> {true}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {false}
learnUnicast: -----> {true}
maxUnicast: -----> {100}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {true}
floodMulticast: -----> {true}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

floodMulticast parameter

The **floodMulticast** parameter allows the MXK to flood all multicast traffic received on a bridge out to all other ports in the VLAN. Multicast traffic in this case usually means multicast traffic that is not for video. For example, many routing protocols are found in multicast packets. This is useful for architectures where the MXK is acting as an aggregation point with no user interfaces. By default, this parameter is set to *true* on TLS bridges and *false* on uplink and downlink bridges.

When set to *true*, this parameter causes all multicast frames to be forwarded out all of the bridge interfaces within the VLAN, except the interface where the multicast was received.

To view the setting for this parameter, enter get **bridge-interface-record**:

```

zSH> bridge add 1-13-1-0/eth tls vlan 500
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth/bridge

zSH> get bridge-interface-record-13-1-0-eth/bridge
bridge-interface-record 1-13-1-0-eth/bridge

```

```

vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {500}
stripAndInsert: -----> {true}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {false}
learnUnicast: -----> {true}
maxUnicast: -----> {100}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {true}
floodMulticast: -----> {true}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

Configure link aggregation bridges

This section describes link aggregation bridging:

- [Create a uplink bridge with link aggregation, page305](#)
- [Create a TLS bridge with link aggregation, page305](#)

Ethernet ports can be bonded together into groups on the Ethernet uplink card or the Active Ethernet card. Link aggregation bridge type is supported on uplink bridges and TLS bridges.

See [Chapter 7, Link Aggregation Configuration](#) for more information on link aggregation.

Create a uplink bridge with link aggregation

Creating an uplink bridge with link aggregation

- 1 Verify link aggregation groups:

```
zSH> linkagg show
```

```
LinkAggregations:
```

slot	unit	ifName	admin	numLinks
a	1	1-a-1-0	up	1
		link: 1-a-2-0 slot a	port 2	subport 0 admin up
b	1	1-b-1-0	up	1
		link: 1-b-2-0 slot b	port 2	subport 0 admin up

- 2 Create an uplink bridge with link aggregation:

```
zSH> bridge add 1-a-1-0/linkagg uplink vlan 333 tagged
```

```
Adding bridge on 1-a-1-0/linkagg
```

```
Created bridge-interface-record 1-a-1-0-linkAgg-333/bridge
```

Deleting the link aggregation bridge

Delete a bridge with link aggregation enter:

```
zSH> bridge delete 1-a-1-0-linkAgg-333/bridge vlan 333
```

```
1-a-1-0-linkAgg-333/bridge Delete complete
```

Create a TLS bridge with link aggregation

Creating a TLS bridge with link aggregation

You can create bridges with link aggregation using both *linkagg* bridge type or *eth* bridge type. If *eth* bridge type is used on an Ethernet port that has been aggregated, the bridge type automatically changes to *linkagg*.

- 1 Create a TLS bridge with link aggregation:

```
zSH> bridge add 1-a-1-0/linkagg tls vlan 777
```

```
Adding bridge on 1-a-1-0/linkagg
```

```
Created bridge-interface-record 1-a-1-0-linkAgg/bridge
```

- 2 To verify the bridge created, enter:

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
tls	777	1-a-1-0-linkAgg/bridge	UP		

- 3 Create a TLS bridge and use the *eth* bridge type:

```
zSH> bridge add 1-a-2-0/eth tls vlan 888
```

```
Adding bridge on 1-a-2-0/eth
```

```
Created bridge-interface-record linkagg-a-1/bridge
```

The bridge type automatically changes to *linkagg*.

4 Verify the bridge created:

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
tls          888 linkagg-a-1/bridge                UP
```

Deleting the link aggregation TLS bridge

To delete the link aggregation bridge if necessary, enter:

```
zSH> bridge delete linkagg-a-1/bridge vlan 888
linkagg-a-1/bridge Delete complete
```

Configure bridge loop issue prevention

Bridge loop issue prevention is configured to resolve certain incorrect MAC address behaviors. The first instance is when the MAC address coming in from the network is then seen as coming from a downlink. The second instance is when the same MAC address is seen by the system as coming from two downlinks at the same time.

Setting the **flapControl** parameter in the **static-bridge** profile to *blockAsym* on an uplink bridge interface on the VLAN ID will block the downlink when incorrect MAC address behavior occurs in a uplink/downlink configuration. When incorrect MAC address behavior involves two downlinks, the bridge interface on the VLAN ID for both downlinks is blocked.

Blocked bridge interfaces must be unblocked with the **bridge unblock interface/type** command.

Configure bridge loop prevention

1 Create the asymmetrical bridging configuration.

Create an uplink bridge.

```
zSH> bridge add 1-a-5-0/eth uplink vlan 700
Adding bridge on 1-a-5-0/eth
Created bridge-interface-record ethernet5-700/bridge
Bridge-path added successfully
```

2 Create the bridge path to enable asymmetrical bridge blocking using **bridge-path add interface/type vlan default flap blockasym**.

```
zSH> bridge-path add ethernet5-700/bridge vlan 700 default flap blockasym
Bridge-path added successfully
```



Note: Enter exactly the same command syntax to enable blocking on an existing bridge path. The existing bridge path will be overwritten, and blocking will be enabled.

3 Create downlink bridges.

```
zSH> bridge add 1-13-1-0/eth downlink vlan 700
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth/bridge
```

```
zSH> bridge add 1-13-2-0/eth downlink vlan 700
Adding bridge on 1-13-2-0/eth
Created bridge-interface-record 1-13-2-0-eth/bridge
```

4 To unblock a bridge that is blocked because of flap protection, use the **bridge unblock interface/type** command.

```
zSH> bridge unblock 1-13-1-0-eth/bridge
```

Dynamic IP filtering on a bridge (Secure DHCP)



Note: MXK uplinks and *network* facing TLS bridges should NOT be configured with a *secure* filter because there are no DHCP client responses possible from network facing bridges. If *secure* is configured on uplink or TLS network facing bridges, traffic will not pass.



Note: For GPON ports, adding *secure* to one VLAN ID will secure the entire port and *all* bridges configured on that port with the same VLAN ID.

The MXK enables secure DHCP settings on downlink bridges, *subscriber* facing TLS bridges, and GPON ports to prevent a user with a statically configured IP address from bypassing DHCP security enforcement. This filter blocks users from accessing the network using anything other than the valid DHCP offered IP address.

When packets are received or sent out a secure downlink bridge interface, TLS subscriber facing bridge interface, or GPON port and VLAN, the MXK checks the IP address against the dynamic IP bridge filter. If a match is found (the address was provided by the DHCP server), the packet is allowed to pass through the filter. Otherwise, it is blocked.

The unicast aging setting for allowed packets is determined based on the DHCP lease time.

Configuring a dynamic IP filter on a bridge

A dynamic IP filter can be configured, modified, and deleted using the bridge add, modify, and delete commands.

- 1 Create a downlink bridge using the **bridge add** command with the **secure** option to create the dynamic IP filter. The **secure** option creates two static bridge paths (MAC and IP) for each host on the bridge that successfully negotiates its IP address from the DHCP server.

```
zSH> bridge add 1-13-9-0/eth downlink vlan 109 slan 509 tagged secure
Adding bridge on 1-13-9-0/eth
Created bridge-interface-record 1-13-9-0-eth-109/bridge
```

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table Data
dwn	Tg 101/501	1-13-1-0-eth-101/bridge	DWN	
dwn	Tg 102/502	1-13-10-0-eth-102/bridge	DWN	
dwn	700	1-13-10-0-eth/bridge	DWN	
dwn	Tg 109/509	1-13-9-0-eth-109/bridge	DWN	

- 2 Display the **bridge-interface-record** for the configured downlink bridge to view the detailed bridge settings.

```
zSH> get bridge-interface-record-13-9-0-eth-101/bridge
```

```
bridge-interface-record 1-13-9-0-eth-101/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {109}
stripAndInsert: -----> {false}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {509}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {mac+ip} secure DHCP enabled
bridgeIfDhcpLearn: -----> {mac+ip} secure DHCP enabled
mvrVlan: -----> {0}
```

- 3 Use the **bridge modify** command to add a dynamic IP filter to an existing bridge or remove a dynamic IP filter from a secure bridge.

```
zSH> bridge modify1-13-9-0/eth secure
```

The **bridgeIfTableBasedFilter** parameter and the **bridgeIfDhcpLearn** parameter in the **bridge-interface-record** profile changes to *mac+ip*.

```
bridgeIfTableBasedFilter: -----> {mac+ip}
bridgeIfDhcpLearn: -----> {mac+ip}
```

```
zSH> bridge modify1-13-9-0/eth non-secure
```

The **bridgeIfTableBasedFilter** parameter and the **bridgeIfDhcpLearn** parameter in the **bridge-interface-record** profile changes to the default *NONE{0}*.

```
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
```

Deleting the dynamic IP filter on a bridge

Delete the dynamic IP on a bridge filter if necessary.

```
zSH> bridge delete1-13-9-0-eth-109/bridge vlan 109 slan 509
1-13-9-0-eth-109/bridge Delete complete
```

Broadcast suppression

Broadcast suppression enables DHCP information to be relayed between DHCP client and host while broadcast filtering is enabled.

The **bridgeifCustomDHCP** setting enables bridge interfaces to pass DHCP information independent of the filterBroadcast setting. Setting **bridgeifCustomDHCP** to *true* will cause that bridge interface to pass DHCP OFFER and ACK packets even though the filterBroadcast is set to *true*.

To enable **bridgeifCustomDHCP** on an existing bridge, update the **bridge-interface-record**.

```
zSH> update bridge-interface-record1-13-1-0-eth-101/bridge
bridge-interface-record 1-13-1-0-eth-101/bridge
Please provide the following: [q]uit.
vpi: -----> {0}:
vci: -----> {0}:
vlanId: -----> {101}:
stripAndInsert: -----> {false}:
customARP: -----> {false}:
filterBroadcast: -----> {false}:
learnIp: -----> {false}:
learnUnicast: -----> {false}:
maxUnicast: -----> {0}:
learnMulticast: -----> {false}:
forwardToUnicast: -----> {false}:
forwardToMulticast: -----> {false}:
forwardToDefault: -----> {true}:
bridgeIfCustomDHCP: -----> {false}true
```

```

bridgeIfIngressPacketRuleGroupIndex: -> {0}:
vlanIdCOS: -----> {0}:
outgoingCOSOption: -----> {disable}:
outgoingCOSValue: -----> {0}:
s-tagTPID: -----> {0x8100}:
s-tagId: -----> {501}:
s-tagStripAndInsert: -----> {true}:
s-tagOutgoingCOSOption: -----> {s-tagdisable}:
s-tagIdCOS: -----> {0}:
s-tagOutgoingCOSValue: -----> {0}:
mcastControlList: -----> {}:
maxVideoStreams: -----> {0}:
isPPPoA: -----> {false}:
floodUnknown: -----> {false}:
floodMulticast: -----> {false}:
bridgeIfEgressPacketRuleGroupIndex: --> {0}:
bridgeIfTableBasedFilter: -----> {NONE(0)}:
bridgeIfDhcpLearn: -----> {NONE(0)}:
mvrVlan: -----> {0}
.....
Save changes? [s]ave, [c]hange or [q]uit:
Record updated.

```

Administrative commands

The MXK provides the following administrative bridging commands:

- **bridge delete**
- **bridge show**
- **bridge showall**
- **bridge-path add**
- **bridge-path show**
- **bridge-path delete**
- **bridge stats**
- **bridge flush**

Refer to the *MALC CLI Reference Guide* for a detailed explanation of the available bridge commands.



Note: Entering general CLI commands on systems with large GPON configurations can take a long time to process. You must be as specific as possible with CLI commands. For example, **bridge flush all** should not be used. Instead, use commands based on the specific interface or MAC address.

Bridge delete command

The **bridge delete** command deletes a specific bridge entry from the system.

Bridge show/showall commands

The **bridge show** and **bridge showall** commands display either a single bridge path entry or the entire bridge table.

```

zSH> bridge showall
Type VLAN          Bridge                               St  Table Data
-----
upl Tagged 998      linkagg-a-1-998/bridge                UP  S VLAN 998 default [U: 3600 sec,
M: 61 sec, I: 30 sec]
tls Tagged 666      1-9-1-710-gponport-666/bridge        UP
tls Tagged 142      linkagg-a-1-142/bridge                UP  D 00:00:0c:07:ac:00
                                         I=516 A=3189 U=2627 F=0
                                         D 00:b0:c2:f2:b6:fc
                                         I=516 A=103330 U=2627 F=0
                                         D 00:b0:c2:f5:54:ee
                                         I=516 A=3189 U=2627 F=0

upl Tagged 1101     1-a-1-0-linkAgg-1101/bridge          UP
upl Tagged 500      linkagg-a-1-500/bridge                UP  S VLAN 500 default [U: 3600 sec,
M: 150 sec, I: 0 sec]
dwn Tagged 998      1-9-1-921-gponport-998/bridge        DWN
dwn Tagged 998      1-9-1-922-gponport-998/bridge        UP
tls Tagged 142      1-9-1-531-gponport-142/bridge        DWN
tls Tagged 3843     1-a-1-0-linkAgg-3843/bridge          UP
upl                134 ethernet4/bridge                      INI NOT LOCAL
tls Tagged 2820     linkagg-a-1-2820/bridge                UP  D 00:1c:15:00:7b:2a
                                         I=553 A=103425 U=2669 F=0
tls Tagged 2920     linkagg-a-1-2920/bridge                UP  D 00:1c:15:00:7b:42
                                         I=554 A=102870 U=2668 F=0
tls Tagged 2820     1-9-1-910-gponport-2820/bridge        UP  D 00:1c:15:00:7b:42
                                         I=555 A=102754 U=2667 F=0
tls Tagged 2920     1-9-1-911-gponport-2920/bridge        UP  D 00:1c:15:00:7b:2a
                                         I=556 A=103425 U=2668 F=0
tls Tagged 2620     linkagg-a-1-2620/bridge                UP  D 00:00:02:00:00:01
                                         I=562 A=4009 U=2653 F=0
tls Tagged 2720     linkagg-a-1-2720/bridge                UP  D 00:00:01:00:00:01
                                         I=566 A=3189 U=2648 F=0

tls Tagged 2720     1-9-1-511-gponport-2720/bridge        UP
tls Tagged 2620     1-9-1-510-gponport-2620/bridge        UP
Aging counter:      48400
Age out counter:    0
Renew failed:       0
Filter renewed:     0
Flap Suppresses:   0
Flap Enabled:       False

zSH> bridge show
Type VLAN          Bridge                               St  Table Data

```

```

-----
upl Tagged 998    linkagg-a-1-998/bridge      UP  S VLAN 998 default [U: 3600 sec,
M: 61 sec, I: 30 sec]
tls Tagged 666    1-9-1-710-gponport-666/bridge  UP
tls Tagged 142    linkagg-a-1-142/bridge      UP  D 00:00:0c:07:ac:00
                                     D 00:b0:c2:f2:b6:fc
                                     D 00:b0:c2:f5:54:ee

upl Tagged 1101   1-a-1-0-linkAgg-1101/bridge  UP
upl Tagged 500    linkagg-a-1-500/bridge      UP  S VLAN 500 default [U: 3600 sec,
M: 150 sec, I: 0 sec]
dwn Tagged 998    1-9-1-921-gponport-998/bridge  DWN
dwn Tagged 998    1-9-1-922-gponport-998/bridge  UP
tls Tagged 142    1-9-1-531-gponport-142/bridge  DWN
tls Tagged 3843   1-a-1-0-linkAgg-3843/bridge  UP
upl              134 ethernet4/bridge          INI NOT LOCAL
tls Tagged 2820   linkagg-a-1-2820/bridge      UP  D 00:1c:15:00:7b:2a
tls Tagged 2920   linkagg-a-1-2920/bridge      UP  D 00:1c:15:00:7b:42
tls Tagged 2820   1-9-1-910-gponport-2820/bridge UP  D 00:1c:15:00:7b:42
tls Tagged 2920   1-9-1-911-gponport-2920/bridge UP  D 00:1c:15:00:7b:2a
tls Tagged 2620   linkagg-a-1-2620/bridge      UP  D 00:00:02:00:00:01
tls Tagged 2720   linkagg-a-1-2720/bridge      UP  D 00:00:01:00:00:01
tls Tagged 2720   1-9-1-511-gponport-2720/bridge UP
tls Tagged 2620   1-9-1-510-gponport-2620/bridge UP

```

Statistics on demand

On the MXK, the statistics are available on demand. You can enable or disable displaying received packet information in the **bridge stats** command. This command enables or disables bridge statistics per port.

There are a total of 256 interfaces on which statistics can be enabled.



Note: The statistics on demand command is currently only available on the ingress ports of GPON cards.

bridge stats enable|disable

The **bridge stats enable|disable** command enables or disables the display of received packet information in the bridge stats command.

Syntax **bridge stats enable|disable** <port #>

Example 1 Displaying bridge stats by port:

```

zSH> bridge stats1-7-7-503-gponport-500
Interface                               Received Packets/Sec  Transmitted Packets/Sec
Name                                   UCast  MCast  BCast  UCast  MCast  BCast  Error
1-7-7-503-gponport-500                 11831   0       0      11837   0      28     0

```

Example 2 Disabling the received packet information:

```

zSH> bridge statsdisable 1-7-7-503-gponport-500/bridge bridge stats
Interface                               Received Packets/Sec  Transmitted Packets/Sec

```


Name	UCast	MCast	BCast	UCast	MCast	Bcast	Error
1-7-7-503-gponport-500	--	--	--	14912	0	38	0

Example 3 Enabling the received packet information:

```
zSH> bridge statsenable 1-7-7-503-gponport-500 bridge stats
```

Interface	Received Packets/Sec			Transmitted Packets/Sec			
Name	UCast	MCast	BCast	UCast	MCast	Bcast	Error
1-7-7-503-gponport-500	18807	0	0	18815	0	49	0

bridge stats list

The **bridge stats list** command displays the list of ports for which **bridge stats** are enabled as well as the summary number of available ports on which **bridge stats** can be enabled

Syntax **bridge stats list**

Example Displaying the list of ports for which bridge stats are enabled:

```
zSH> bridge stats list
Processing list of 134
1-7-7-503-gponport-500
```

bridge stats rules

The **bridge stats rules** command displays a summary of the interfaces which are in use (called rules) and the remaining ingress rules on a per slot basis.

Syntax **bridge stats rules**

Example This example displays one interface displaying bridge statistics, so there are 255 or the total 256 possible *rules* (per port) available.

```
zSH> bridge stats rules
```

Processing list of 134

Slot	Total Rules In Use	Total Rules Remaining
====	=====	=====
7	1	255

6 VIDEO CONFIGURATION

This chapter explains how to configure the MXK for video and includes the following sections:

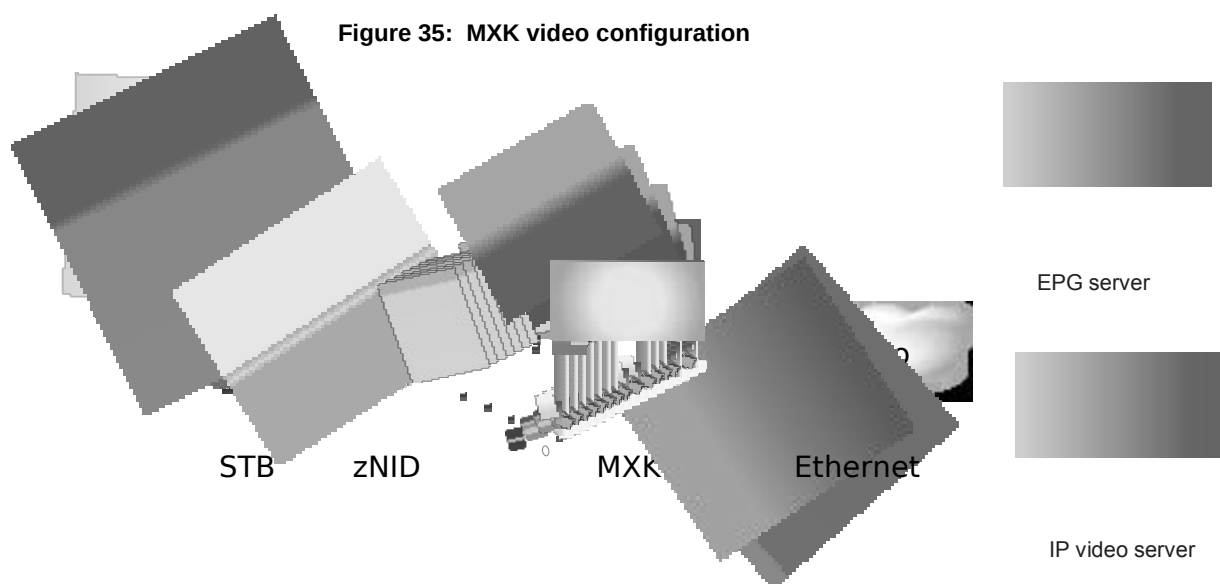
- [MXK routed video, page315](#)
- [Configure routed video connections on the MXK, page316](#)
- [MXK bridged video, page331](#)
- [Configure bridged video on the MXK, page332](#)
- [IGMP snooping with proxy reporting, page340](#)

MXK routed video

When configuring an interface for IP video, you need to dedicate a VLAN on the logical interface so that IP video can be delivered to the subscriber. This is because using the same VLAN to transmit other types of traffic, such as voice or data, could affect the quality of the video delivery.

For bridged video, see [MXK bridged video on page331](#).

[Figure35](#) shows a MXK video configuration.



Configure routed video connections on the MXK

This section describes how to configure the MXK for IP video connections:

- [Routed video configuration overview, page316](#)
- [Configure host-based routing for video with local DHCP, page316](#)
- [Configure host-based routing for video with dhcp-relay agent\(s\), page322](#)

Routed video configuration overview

Generally these are the steps to follow to configure the MXK for routed video.

Configuring the MXK for routed video

- 1 Create an IP interface on an uplink Ethernet port.
See [Configuring an IP interface on an uplink Ethernet port on page317](#).
- 2 Create a video source connection between the IP interface and the multi-cast address.
See [Creating the video source profile on page317](#).
- 3 Create a floating IP interface for communication between the MXK, video source, hosts and DHCP server.
See [Creating a floating IP interface on page318](#).
- 4 Create a **dhcp-server-subnet** profile referencing the floating IP interface to run DHCP locally on the MXK.
See [Creating a dhcp-server-subnet on page319](#).
Or
Create the DHCP server address pool referencing the floating IP interface with **dhcp-relay add** for an external DHCP server configuration.
See [Adding the dhcp-relay agent on page325](#)
- 5 Create the multicast control lists.
See [Creating multicast control lists on page328](#).
- 6 Add a host route for the downstream devices.
See [Adding the host routes on Ethernet interfaces on page329](#).

Configure host-based routing for video with local DHCP

This configuration procedure uses the MXK to provide local DHCP services.

Configuring an IP interface on an uplink Ethernet port

Configuring the MXK for video requires that first you add an IP interface with a VLAN to an uplink Ethernet port, then create a video source connection between that IP interface and the multicast address by creating a **video-source** profile.

Create an IP interface on the uplink port:

Create an IP interface on the MXK 10 GE port with VLAN ID 999 for the IP video:

```
zSH> interface add 1-a-2-0/eth vlan 999 192.169.1.14/24
Created ip-interface-record ethernet2-999/ip.
```

Verify the interface:

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address          Media/Dest Address  IfName
-----
1/a/1/0/ip     UP      1 172.16.160.49/24  00:01:47:17:ee:54   ethernet1
1/a/2/0/ip     DOWN    1 192.169.1.14/24   00:01:47:17:ee:55   ethernet2-999
-----
```

Creating the video source profile

Create a mapping between the video connection and the multicast address space. The **video-source** profile specifies the interface the MXK uses to reach the IP video server. (The following example uses the uplink interface *ethernet2-999* to reach the IP video server). Multisource multicast enables IGMP join/leaves to the video headend for each configured **video-source** profile. One **video-source** profile is assigned to each uplink IP interface.

The convention for multicast addresses is that they begin with 244.

1 Create the **video-source** profile.

```
zSH> new video-source 1
video-source 1
Please provide the following: [q]uit.
routing-domain: ----> {0}: 1
multicast-address: -> {0.0.0.0}: 224.1.1.1
ifIndex: -----> {0/0/0/0/0}: ethernet2-999/ip
vpi: -----> {0}:
vci: -----> {0}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```



Note: You only need to enter the first multicast address in the group.

Or

Use the **videosource add** command to create the video source connection:

```
zSH> videosource adddomain 1 224.1.1.1 ethernet2-999/ip
Added video-source profile
```

2 View the **video-source** profile:

```
zSH> get video-source 1
video-source 1
routing-domain: ----> {1}
multicast-address: -> {224.1.1.1}
ifIndex: -----> {ethernet2-999/ip}
vpi: -----> {0}
vci: -----> {0}
```

Creating a floating IP interface

You must create a floating IP interface to provide video set-top boxes a range of IP addresses for their far-end address.

1 Create a floating IP interface.

You can designate an index name such as *video* for the floating IP interface.

```
zSH> interface add floatvideo 10.107.8.254 255.255.255.0
Created ip-interface-record video/ip.
```

The **ip-interface-record** profile is created with the IP address, the subnet, and the broadcast address.

2 Verify that the interface was created:

```
zSH> list ip-interface-record
ip-interface-record ethernet1/ip
ip-interface-record ethernet2-999/ip
ip-interface-record video/ip
3 entries found.
```

3 Verify the **ip-interface-record** profile for the floating IP video interface, *video/ip*:

```
zSH> get ip-interface-record video/ip
ip-interface-record video/ip
vpi: -----> {0}
vci: -----> {0}
rdindex: -----> {1}
dhcp: -----> {none}
addr: -----> {10.107.8.254}
netmask: -----> {255.255.255.0}
bcastaddr: -----> {10.107.8.255}
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmaxsize: -----> {0}
```

```

ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: -> {0}
egressPacketRuleGroupIndex: --> {0}
ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {false}
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {999}
maxVideoStreams: -----> {0}
tosOption: -----> {disable}
tosCOS: -----> {0}
vlanCOS: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}

```

Creating a dhcp-server-subnet

You need to create a **dhcp-server-subnet** profile to reference the floating IP interface and create the basis for local DHCP on the MXK.

- 1 Create the **dhcp-server-subnet** profile by entering the floating IP interface in the **dhcp-server-subnet** profile parameters.

```

zSH> new dhcp-server-subnet 1
dhcp-server-subnet 1
Please provide the following: [q]uit.
network: -----> {0.0.0.0}: 10.107.8.0
netmask: -----> {0.0.0.0}: 255.255.255.0
domain: -----> {0}:
range1-start: -----> {0.0.0.0}: 10.107.8.1
range1-end: -----> {0.0.0.0}: 10.107.8.250
range2-start: -----> {0.0.0.0}:
range2-end: -----> {0.0.0.0}:
range3-start: -----> {0.0.0.0}:
range3-end: -----> {0.0.0.0}:
range4-start: -----> {0.0.0.0}:
range4-end: -----> {0.0.0.0}:
default-lease-time: ----> {-1}:
min-lease-time: -----> {-1}:
max-lease-time: -----> {-1}:
boot-server: -----> {0.0.0.0}:
bootfile: -----> {}:
default-router: -----> {0.0.0.0}: 10.107.8.254
primary-name-server: ---> {0.0.0.0}:
secondary-name-server: -> {0.0.0.0}:

```

```

domain-name: -----> {}:
subnetgroup: -----> {0}: 1
stickyaddr: -----> {enable}:
external-server: -----> {0.0.0.0}:
external-server-alt: ---> {0.0.0.0}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

```

2 View the **dhcp-server-subnet** profile.

```

zSH> get dhcp-server-subnet 1
dhcp-server-subnet 1
network: -----> {10.107.8.0}
netmask: -----> {255.255.255.0}
domain: -----> {0}
range1-start: -----> {10.107.8.1}
range1-end: -----> {10.107.8.250}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: ----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {10.107.8.254}
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {1}
stickyaddr: -----> {enable}
external-server: -----> {0.0.0.0}
external-server-alt: ---> {0.0.0.0}

```

Creating a multicast control list

Create a multicast control list to define which multicast addresses the remote-end video can access. A multicast control list entry of 0 enables subscriptions up to the number of maximum video streams on the interface without control list checking.

1 The following example adds three entries to multicast list 1:

```

zSH> new mcast-control-entry 1/1
mcast-control-entry 1/1
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.1
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s

```


New record saved.

```
zSH> new mcast-control-entry 1/2
mcast-control-entry 1/2
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.2
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

```
zSH> new mcast-control-entry 1/3
mcast-control-entry 1/3
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.3
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Continue adding as many multicast entries as necessary.

2 Verify the multicast entries:

```
zSH> mcast show mcl1
MCAST CONTROL LIST : 1
                     224.1.1.1           224.1.1.2           224.1.1.3
```

Adding a host route on an Ethernet interface

Add a host route with a VLAN ID for the video interface on the card facing the subscriber. The *1* references the **dhcp-server-subnet** group and the *5* is the number of floating IP addresses the system will allow. Enter the keyword *video* to set which multicast control list is used and how many of video streams are allowed on the interface.

1 Add the host to the Ethernet interface.

```
zSH> host add 1-13-1-0/eth/vlan 200 dynamic 1 5 video 2/10
Adding host for 1-13-1-0/eth
```

2 View the host.

```
zSH> host show
```

Rd/Address	Interface	Group	T	Host	Address
1 10.107.8.254	1-13-1-0-eth-200	1	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	

Deleting the configuration

When necessary, you can delete the host-based routing configuration for video.

- 1 Delete the host.

```
zSH> host delete 1-13-1-0/eth vlan 200 all
Deleting host for 1-13-1-0/eth
```

- 2 Delete the **dhcp-server-subnet** profile, group 1.

```
zSH> delete dhcp-server-subnet 1
dhcp-server-subnet 1
1 entry found.
Delete dhcp-server-subnet 1? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 1 deleted.
```

- 3 Delete the floating IP interface named *video*.

```
zSH> interface delete float video
Interface video deleted
```

- 4 Delete the multicast control lists.

```
zSH> delete mcast-control-entry 1/1
mcast-control-entry 1/1
1 entry found.
Delete mcast-control-entry 1/1? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/1 deleted.
```

```
zSH> delete mcast-control-entry 1/2
mcast-control-entry 1/2
1 entry found.
Delete mcast-control-entry 1/2? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/2 deleted.
```

```
zSH> delete mcast-control-entry 1/3
mcast-control-entry 1/3
1 entry found.
Delete mcast-control-entry 1/3? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/3 deleted.
```

- 5 Delete the video source connection.

```
zSH> videosource deletedomain 1 224.1.1.1 ethernet2-999/ip
Deleted video-source profile
```

Configure host-based routing for video with dhcp-relay agent(s)

This procedure configures the MXK for video services using a DHCP server in the network.

Configuring an IP interface on an uplink Ethernet port

Configuring the MXK for video requires that first you add an IP interface with a VLAN to an uplink Ethernet port, then create a video source connection between that IP interface and the multicast address by creating a **video-source** profile.

Create an IP interface on the uplink port:

Create an IP interface on the MXK GigaBit Ethernet port with VLAN ID 999 for the IP video:

```
zSH> interface add 1-a-2-0/eth vlan 999 192.169.1.14/24
Created ip-interface-record ethernet2-999/ip.
```

Verify the interface:

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address          Media/Dest Address  IfName
-----
1/a/1/0/ip     UP      1 172.16.160.49/24  00:01:47:17:ee:54   ethernet1
1/a/2/0/ip     DOWN    1 192.169.1.14/24   00:01:47:17:ee:55   ethernet2-999
-----
```

Creating the video source profile

Create a mapping between the video connection and the multicast address space. The **video-source** profile specifies the interface the MXK uses to reach the IP video server. (The following example uses the uplink interface *ethernet2-999* to reach the IP video server). Multisource multicast enables IGMP join/leaves to the video headend for each configured **video-source** profile. One **video-source** profile is assigned to each uplink IP interface.

The convention for multicast addresses is that they begin with 244.

1 Create the **video-source** profile.

```
zSH> new video-source 1
video-source 1
Please provide the following: [q]uit.
routing-domain: ----> {0}: 1
multicast-address: -> {0.0.0.0}: 224.1.1.1
ifIndex: -----> {0/0/0/0/0}: ethernet2-999/ip
vpi: -----> {0}:
vci: -----> {0}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```



Note: You only need to enter the first multicast address in the group.

Or

Use the **videosource add** command to create the video source connection:

```
zSH> videosource adddomain 1 224.1.1.1 ethernet2-999/ip
Added video-source profile
```

2 View the **video-source** profile:

```
zSH> get video-source 1
video-source 1
routing-domain: ----> {1}
multicast-address: -> {224.1.1.1}
ifIndex: -----> {ethernet2-999/ip}
vpi: -----> {0}
vci: -----> {0}
```

Creating a floating IP interface

You must create a floating IP interface to provide video set-top boxes a range of IP addresses for their far-end address.

1 Create a floating IP interface.

You can designate an index name, such as *flt1*, for the floating IP interface.

```
zSH> interface add float flt1 10.107.8.254 255.255.255.0
Created ip-interface-record flt1/ip.
```

The **ip-interface-record** profile is created with the IP address, the subnet, and the broadcast address.

2 Verify that the interface was created:

```
zSH> list ip-interface-record
ip-interface-record ethernet1/ip
ip-interface-record ethernet2-999/ip
ip-interface-record flt1/ip
3 entries found.
```

3 Verify the **ip-interface-record** profile for the floating IP video interface, *video/ip*:

```
zSH> get ip-interface-record flt1/ip
ip-interface-record flt1/ip
vpi: -----> {0}
vci: -----> {0}
rdindex: -----> {1}
dhcp: -----> {none}
addr: -----> {10.107.8.254}
netmask: -----> {255.255.255.0}
bcastaddr: -----> {10.107.8.255}
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmaxsize: -----> {0}
```

```

ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: -> {0}
egressPacketRuleGroupIndex: --> {0}
ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {false}
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {0}
maxVideoStreams: -----> {0}
tosOption: -----> {disable}
tosCOS: -----> {0}
vlanCOS: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}

```

Adding the dhcp-relay agent

You must create a DHCP server address pool for the far-end video set-top device by using the **dhcp-relay add** command to create the dhcp-relay agent. The subnet address/mask is derived from the floating IP address to provide the address pool.

The syntax for the **dhcp-relay add** command is:

dhcp-relay add

```

Usage: dhcp-relay <add|delete|modify|show>
[<subnetgroup>] <ip-address> [alt <ip-address>]
[<interface>/<type> | NULL]

```

- 1 Create the **dhcp-server-subnet** by entering the external DHCP server IP address and the index name of the floating IP interface.

```

zSH> dhcp-relay add 102.168.88.73 flt1
Created DHCP Relay Agent number 1

```

- 2 Verify the dhcp-relay agent and the agent number:

```

zSH> list dhcp-server-subnet
dhcp-server-subnet 2
dhcp-server-subnet 1
2 entries found.

```

- 3 View the **dhcp-server-subnet** profile created with the **dhcp-relay add** command.

```

zSH> get dhcp-server-subnet 1
dhcp-server-subnet 1
network: -----> {10.107.8.0}  floating IP address
netmask: -----> {255.255.255.0}  floating IP address subnet
domain: -----> {0}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: ----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {10.107.8.254}  references the floating IP address
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {1}  subnet group of the dhcp-server-subnet
stickyaddr: -----> {enable}
external-server: -----> {102.168.88.73}  IP address of the external DHCP server
external-server-alt: ---> {0.0.0.0}

```

Creating multiple floating IP interfaces and multiple dhcp-relay agents

You can create more than one group of floating IP addresses on the MXK. After creating more than one floating IP interfaces, you can use the **dhcp-relay add** command to create several **dhcp-server-subnet** groups and choose which floating IP interface to associate with the **dhcp-server-subnet** group.

1 Create the floating IP interfaces:

```

zSH> interface add float 172.25.45.254 255.255.255.0
Created ip-interface-record 172.25.45.254/ip.

```

```

zSH> interface add float 172.25.46.254 255.255.255.0
Created ip-interface-record 172.25.46.254/ip.

```

2 Create the dhcp-server relay agent by designating the IP address of the DHCP server and the floating interface:

```

zSH> dhcp-relay add 172.24.8.1 172.25.45.254/ip
Created DHCP Relay Agent number 2

```

In this command example, the relay agent number, 2, is created by the system. You can also designate your own number as follows:

```

zSH> dhcp-relay add 13 172.24.8.1 172.25.46.254/ip

```

Created DHCP Relay Agent number 13

3 Verify the **dhcp-server-subnet** interfaces:

zSH> **get dhcp-server-subnet 2**

```
dhcp-server-subnet 2
network: -----> {172.25.45.0} floating IP interface
netmask: -----> {255.255.255.0} floating IP interface subnet
domain: -----> {}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: -----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {172.25.45.254} floating IP interface
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {2}
stickyaddr: -----> {enable}
external-server: -----> {172.24.8.1} external DHCP server IP address
external-server-alt: ---> {0.0.0.0}
```

zSH> **get dhcp-server-subnet 13**

```
dhcp-server-subnet 13
network: -----> {172.25.46.0} floating IP interface
netmask: -----> {255.255.255.0} floating IP interface subnet
domain: -----> {}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: -----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {172.25.46.254} floating IP interface
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {13}
```

```

stickyaddr: -----> {enable}
external-server: -----> {172.24.8.1}  external DHCP server IP address
external-server-alt: ---> {0.0.0.0}

```

Adding a dhcp-relay agent with DHCP server and alternate DHCP server

You can use the **dhcp-relay add** command to designate the DHCP server and an alternate DHCP server:

- 1 Add the dhcp-relay agent with the DHCP server IP address and the IP address for the alternate DHCP server and the IP interface for the floating IP.

```

zSH> dhcp-relay add3 172.24.8.1 alt 172.24.8.2 172.25.46.254/ip
Created DHCP Relay Agent number 3

```

- 2 Verify the **dhcp-server-subnet**:

```

zSH> get dhcp-server-subnet3
dhcp-server-subnet 3
network: -----> {172.25.46.0} floating IP interface
netmask: -----> {255.255.255.0} floating IP interface subnet
domain: -----> {}
range1-start: -----> {0.0.0.0}
range1-end: -----> {0.0.0.0}
range2-start: -----> {0.0.0.0}
range2-end: -----> {0.0.0.0}
range3-start: -----> {0.0.0.0}
range3-end: -----> {0.0.0.0}
range4-start: -----> {0.0.0.0}
range4-end: -----> {0.0.0.0}
default-lease-time: ----> {-1}
min-lease-time: -----> {-1}
max-lease-time: -----> {-1}
boot-server: -----> {0.0.0.0}
bootfile: -----> {}
default-router: -----> {172.25.46.254} floating IP interface
primary-name-server: ---> {0.0.0.0}
secondary-name-server: -> {0.0.0.0}
domain-name: -----> {}
subnetgroup: -----> {3}
stickyaddr: -----> {enable}
external-server: -----> {172.24.8.1} external DHCP server IP address
external-server-alt: ---> {172.24.8.2} alternate external DHCP server IP address

```

Creating multicast control lists

Create a multicast control list, which defines which multicast addresses the remote-end video can access. A multicast control list entry of 0 enables subscriptions up to the number of maximum video streams on the interface without control list checking. You can configure video streams on Ethernet and GPON interfaces.

1 The following example adds three entries to multicast list 1:

```
zSH> new mcast-control-entry 1/1
mcast-control-entry 1/1
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.1
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

```
zSH> new mcast-control-entry 1/2
mcast-control-entry 1/2
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.2
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

```
zSH> new mcast-control-entry 1/3
mcast-control-entry 1/3
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.3
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Continue adding as many multicast entries as necessary.

2 Verify the multicast entries:

```
zSH> mcast show mcl1
MCAST CONTROL LIST : 1
                224.1.1.1                224.1.1.2                224.1.1.3
```

Adding the host routes on Ethernet interfaces

Add the host routes with VLAN IDs for the video on the interfaces of the card facing the subscriber.

1 Add the hosts.

a Add a host to the Ethernet interface with VLAN ID.

In this case, the 2 references the **dhcp-server-subnet** profile group 2 and the 3 defines number of floating IP addresses allowed on the interface. The key word *video* set the interface for video service. The 1 references the multicast control list and the 5 defines the maximum number of video streams allowed on the interface.

```
zSH> host add 1-13-2-0/eth vlan 101 dynamic 2 3 video 1/5
Adding host for 1-13-2-0/eth
```

b Add a host to the Ethernet interface.

In this case, the 13 references the **dhcp-server-subnet** profile group 13 and the 5 defines number of floating IP addresses allowed on the interface. The key word *video* set the interface for video service. The 2 references the multicast control list and the 5 defines the maximum number of video streams allowed on the interface.

```
zSH> host add 1-13-3-0/eth vlan 201 dynamic 13 5 video 2/5
Adding host for 1-13-3-0/eth
```

c Add a host to the Ethernet interface.

In this case, the 3 references the **dhcp-server-subnet** profile group 3 and the 3 defines number of floating IP addresses allowed on the interface. The key word *video* set the interface for video service. The 2 references the multicast control list and the 5 defines the maximum number of video streams allowed on the interface.

```
zSH> host add 1-13-4-0/eth vlan 301 dynamic 3 3 video 3/5
Adding host for 1-13-4-0/eth
```

2 Verify the hosts:

```
zSH> host show
```

Rd/Address	Interface	Group	T	Host	Address
1 172.25.45.254	1-13-2-0-eth-101	2	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
1 172.25.46.254	1-13-3-0-eth-201	13	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	
1 172.25.46.254	1-13-4-0-eth-301	3	D	<unassigned>	
			D	<unassigned>	
			D	<unassigned>	

Deleting the host-based video configuration

When necessary, you can delete the host-based video configuration.

1 Delete the Ethernet interfaces.

```
zSH> host delete 1-13-2-0/eth vlan 101 all
Deleting host for 1-13-2-0/eth
zSH> host delete 1-13-3-0/eth vlan 201 all
Deleting host for 1-13-3-0/eth
zSH> host delete 1-13-4-0/eth vlan 301 all
Deleting host for 1-13-4-0/eth
```

2 Delete the multicast control lists.

```
zSH> delete mcast-control-entry 1/1
```

```
mcast-control-entry 1/1
1 entry found.
Delete mcast-control-entry 1/1? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/1 deleted.
```

```
zSH> delete mcast-control-entry 1/2
mcast-control-entry 1/2
1 entry found.
Delete mcast-control-entry 1/2? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/2 deleted.
```

```
zSH> delete mcast-control-entry 1/3
mcast-control-entry 1/3
1 entry found.
Delete mcast-control-entry 1/3? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/3 deleted.
```

3 Delete the **dhcp-server-subnet** groups.

```
zSH> delete dhcp-server-subnet 3
dhcp-server-subnet 3
1 entry found.
Delete dhcp-server-subnet 3? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 3 deleted.
zSH> delete dhcp-server-subnet 13
dhcp-server-subnet 13
1 entry found.
Delete dhcp-server-subnet 13? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 13 deleted.
zSH> delete dhcp-server-subnet 2
dhcp-server-subnet 2
1 entry found.
Delete dhcp-server-subnet 2? [y]es, [n]o, [q]uit : y
dhcp-server-subnet 2 deleted.
```

4 Delete the floating IP interfaces.

```
zSH> interface delete float 172.25.45.254
Interface 172.25.45.254 deleted
zSH> interface delete float 172.25.46.254
Interface 172.25.46.254 deleted
zSH> interface delete float flt1
Interface flt1 deleted
```

5 Delete the video source connection.

```
zSH> videosource delete domain 1 224.1.1.1 ethernet2-999/ip
Deleted video-source profile
```

MXK bridged video

Video bridging enables video packets to be forwarded over bridges from a headend device down to downstream device. In this case, the video travels

from the source, or head-end device, using one video stream to passively traverse the MXK backplane. This lowers the bandwidth requirements for video packets traversing the MXK.

Video bridging requires configuring an uplink bridge and a downlink bridge. On the uplink bridge, the **forwardToMulticast** function is associated with a location that contains the video content that allows the MXK to receive video streams from the network. An interface with this value set to *true* only transmits multicast traffic for which a JOIN request was received. A bridge interface with the **forwardToMulticast** parameter set to *false* discards multicast traffic from that interface. By default, the **forwardToMulticast** parameter is set to *true* on uplink bridges and *false* on downlink bridges.

On the downlink bridge, the **learnMulticast** function is associated with interfaces that have hosts connected to them and allows the MXK to send video groups from downlink interfaces to the network. By default, the **learnMulticast** parameter is set to *true* on downlink bridges.

Note that JOIN requests enter on a **learnMulticast** interface associated with a downlink bridge and pass through on a **forwardToMulticast** interface associated with an uplink bridge.

[Table9](#) details various video bridge behaviors associated with different combinations of settings for the bridge parameters.

Table 9: learnMulticast-forwardToMulticast combinations and behavior

learnMulticast	forwardToMulticast	Behavior
False	False	The interface discards all incoming multicast packets and does not forward any of the packets.
True	False	The interface forwards both default multicast signaling packets and control multicast packets.
False	True	The interface forwards control packets received on this interface to all other interfaces that have the learnMulticast field set to true that have sent a JOIN message for a group
True	True	Treat the same as an interface with the learnMulticast field set to false and the forwardToMulticast field set to true .

Configure bridged video on the MXK

This section describes how to configure the MXK for video connections so that traffic passes between the MXK, the upstream video source, and the subscriber:

- [Bridged video connection overview, page333](#)
- [Configure a video connection on the MXK, page333](#)

Bridged video connection overview

Bridged video connections require bridge configurations on the uplink and on the downlink.

Generally, these are the steps to follow to configure the MXK for bridged video.

Configuring the MXK for bridged video

- 1 Create an uplink bridge on a 100/1000 Ethernet port or a 10 GE port on the uplink card.
 - a Create an uplink bridge with a VLAN ID.
See [Creating an uplink bridge on a 100/1000 Ethernet or a 10 GE uplink port for video on page333](#).
 - b Create the bridge path for the uplink bridge with VLAN ID and enter the multicast aging period and the IGMP query interval.
See [Creating an uplink bridge on a 100/1000 Ethernet or a 10 GE uplink port for video on page333](#).
- 2 Create the multicast control lists.
See [Creating multicast control lists on page335](#).
- 3 Create a downlink bridge with a VLAN ID and specify the maximum number of video streams and a multicast control list.
See [Creating a downlink bridge on an Active Ethernet port for video services on page336](#) or [Creating a downlink bridge on a GPON port for video services on page337](#).

Configure a video connection on the MXK

You must create an uplink bridge on a 100/1000 Ethernet or a 10 GE port and configure the bridge for video service and then create a downlink bridge to the subscriber.

Creating an uplink bridge on a 100/1000 Ethernet or a 10 GE uplink port for video

You create a video bridge on the uplink by first creating an uplink bridge on a 100/1000 Ethernet or a 10 GE port with the **bridge add** command using a VLAN ID. Then enter the multicast aging period and IGMP query interval for video traffic when entering the **bridge-path add** command.

- 1 Create a tagged uplink bridge on a 100/1000 Ethernet or a 10 GE port with a VLAN ID. Designating *tagged* will pass the VLAN ID up to the network.

```
zSH> bridge add 1-a-4-0/eth uplink vlan 101 tagged
Adding bridge on 1-a-4-0/eth
```

Created bridge-interface-record ethernet4-101/bridge

View the **bridge-interface-record** profile:

Specifying *uplink* sets the **forwardToMulticast** parameter to *true* and the **learnMulticast** parameter to *false*. The **bridge-interface-record** is configured to send multicast packets to interfaces that send a JOIN request.

Specifying *tagged* sets the **stripAndInsert** and **s-tagStripAndInsert** parameters to *false*. All packets with VLAN ID 101 will pass through the uplink interface to the network intact.

```
zSH> get bridge-interface-record ethernet4-101/bridge
bridge-interface-record ethernet4-101/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {101}
stripAndInsert: -----> {false} VLAN ID is passed on to the network
customARP: -----> {true}
filterBroadcast: -----> {true}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {true} Forwards video streams to interfaces that send a JOIN request
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {true}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}
```

- 2 Add the bridge path and set the multicast aging period and IGMP query interval.

The **mcast** sets the maximum age, in seconds, of a multicast packet before it is purged

The **igmptimer** indicates a time value in seconds. This value should be greater than 0. If you enter 0, the querying function is disabled.

```
zSH> bridge-path addethernet4-101/bridge vlan 101 default mcast 90 igmptimer 30
Bridge-path added successfully
```

3 Verify the bridge and bridge path.

```
zSH> bridge show
Type VLAN      Bridge
-----
upl Tagged 101  ethernet4-101/bridge
UP   S VLAN 101 default

zSH> bridge-path show
VLAN/SLAN      Bridge      Address
-----
101 ethernet4-101/bridge  Default
```

Creating multicast control lists

Create a multicast control list, which defines which multicast addresses the remote-end video can access. A multicast control list entry of 0 enables subscriptions up to the number of maximum video streams on the interface without control list checking. You can configure video streams on Ethernet and GPON interfaces.

1 The following example adds three entries to multicast list 1:

```
zSH> new mcast-control-entry 1/1
mcast-control-entry 1/1
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.1
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

```
zSH> new mcast-control-entry 1/2
mcast-control-entry 1/2
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.2
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

```
zSH> new mcast-control-entry 1/3
mcast-control-entry 1/3
Please provide the following: [q]uit.
ip-address: -> {0.0.0.0}: 224.1.1.3
type: -----> {normal}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

Continue adding as many multicast entries as necessary.

2 Verify the multicast entries:

```
zSH> mcast show mcl1
MCAST CONTROL LIST : 1
                     224.1.1.1           224.1.1.2           224.1.1.3
```

Creating a downlink bridge on an Active Ethernet port for video services

You can create a downlink bridge on an Active Ethernet port with a VLAN ID. You can also specify a maximum number of video streams and a multicast control list. Add the multicast control list and designate the maximum video streams using the *m/n* format. The multicast control list is set first and the maximum video streams second.

Members of the multicast control list must be defined to receive the video signal.

Create a downlink bridge with VLAN ID on an Active Ethernet port.

```
zSH> bridge add 1-13-1-0/eth downlink vlan 101 tagged video 1/6
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth-101/bridge
```

Verify the bridge.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
dwn	Tagged 101	1-13-1-0-eth-101/bridge	DWN		
upl	Tagged 101	ethernet4-101/bridge	UP	S	VLAN 101 default

Specifying *downlink* sets the **learnMulticast** parameter to *true* and the **forwardToMulticast** parameter to *false*. When the **learnMulticast** parameter is *true*, this allows multicast packets to pass to the subscriber after a JOIN request is sent.

Specifying *tagged* sets the **stripAndInsert** parameter to *false* causing the VLAN ID to pass to the downstream device.

View the **bridge-interface-record** profile.

```
zSH> get bridge-interface-record-13-1-0-eth-101/bridge
bridge-interface-record 1-13-1-0-eth-101/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {101}
stripAndInsert: -----> {false}
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true}
```



```

forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false}
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

Creating a downlink bridge on a GPON port for video services

Create a downlink bridge on a GPON port by first creating a GPON traffic descriptor and then configuring the downlink bridge.

- 1 Create the GPON traffic descriptor.

```

zSH> new gpon-traffic-profile 1
gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}:
traffic-class: -----> {ubr}:
compensated: -----> {false}:
shared: -----> {false}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

```

- 2 Configure a downlink bridge on a GPON port for video.

You create a downlink bridge on an GPON port with VLAN ID. You can also specify a maximum number of video streams and a multicast control list. Add the multicast control list and designate the maximum video streams using the *m/n* format. The multicast control list is set first and the maximum video streams second.

Members of the multicast control list must be defined to receive the video signal.

```

zSH> bridge add1-9-1-502/gponport gtp 1 downlink vlan 101 tagged video 1/3

```

GEM Port 1-9-1-502/gponport has been created on ONU 1-9-1-2/gpononu.
 Adding bridge on 1-9-1-502/gponport
 Created bridge-interface-record 1-9-1-502-gponport-101/bridge

Verify the bridge interface.

zSH> **bridge show**

Type	VLAN	Bridge	St	Table	Data
dwn	Tagged 101	1-13-1-0-eth-101/bridge	DWN		
dwn	Tagged 101	1-9-1-502-gponport-101/bridge	DWN		
upl	Tagged 101	ethernet4-101/bridge	UP	S	VLAN 101 default

Specifying *downlink* sets the **learnMulticast** parameter to *true* and the **forwardToMulticast** parameter to *false*. When the **learnMulticast** parameter is *true*, multicast packets are allowed to pass to the subscriber after a JOIN request is sent.

Specifying *tagged* sets the **stripAndInsert** parameter to *false* causing the VLAN ID to pass to the downstream device.

View the **bridge-interface-record** profile.

zSH> **get bridge-interface-record-9-1-502-gponport-101/bridge**

```
bridge-interface-record 1-9-1-502-gponport-101/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {101}
stripAndInsert: -----> {false} VLAN 101 is passed down to the device
customARP: -----> {false}
filterBroadcast: -----> {false}
learnIp: -----> {true}
learnUnicast: -----> {true}
maxUnicast: -----> {5}
learnMulticast: -----> {true} multicast packets will not be discarded
forwardToUnicast: -----> {false}
forwardToMulticast: -----> {false} network will not accept multicast packets from this interface
forwardToDefault: -----> {true}
bridgeIfCustomDHCP: -----> {false}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControlList: -----> {1}
maxVideoStreams: -----> {3}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
```

```
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}
```

Deleting the video configuration

If necessary, you can delete the uplink bridge, bridge path, multicast control lists, and downlink bridges.

1 Delete the multicast control lists.

```
zSH> delete mcast-control-entry 1/1
mcast-control-entry 1/1
1 entry found.
Delete mcast-control-entry 1/1? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/1 deleted.
```

```
zSH> delete mcast-control-entry 1/2
mcast-control-entry 1/2
1 entry found.
Delete mcast-control-entry 1/2? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/2 deleted.
```

```
zSH> delete mcast-control-entry 1/3
mcast-control-entry 1/3
1 entry found.
Delete mcast-control-entry 1/3? [y]es, [n]o, [q]uit : y
mcast-control-entry 1/3 deleted.
```

2 Delete the Active Ethernet downlink bridge.

```
zSH> bridge delete 1-13-1-0-eth-101/bridge
1-13-1-0-eth-101/bridge Delete complete
```

3 Delete the GPON traffic profile.

Optional. You do not need to delete this profile to delete the GPON downlink bridge and the profile may be used by other bridges.

```
zSH> delete gpon-traffic-profile 1
gpon-traffic-profile 1
1 entry found.
Delete gpon-traffic-profile 1? [y]es, [n]o, [q]uit : y
gpon-traffic-profile 1 deleted.
```

4 Delete the GPON downlink bridge.

```
zSH> bridge delete 1-9-1-502-gponport-101/bridge
1-9-1-502-gponport-101/bridge Delete complete
GEM Port 1-9-1-502/gponport has been deleted.
```

5 Delete the bridge path for the uplink bridge.

```
zSH> bridge-path delete ethernet4-101/bridge vlan 101 default
Delete complete
```

6 Delete the uplink bridge.

```
zSH> bridge delete ethernet4-101/bridge
ethernet4-101/bridge Delete complete
```

IGMP snooping with proxy reporting

This section describes:

- [IGMP snooping overview, page340](#)
- [Join requests, page340](#)
- [Leave requests, page341](#)
- [IGMP snooping with proxy configuration, page341](#)
- [IGMPv3 messages respond for STBs, page346](#)

IGMP snooping overview

IGMP snooping applies to bridged video. Enabling IGMP snooping reduces traffic between the MXK and the upstream multicast headend device by changing the behavior of the MXK for more efficient tracking and grouping of JOIN and LEAVE requests. MXK IGMP snooping also supports the following:

- Solicited or unsolicited query reports.
- Ability to configure the MXK to send queries to hosts; by default the MXK does not.
- Queries are sent only to hosts that have sent a join request.
- Compliance with rfc4541 regarding IGM forwarding and data rules.
- Information table is available during redundant uplink card switchovers.
- Membership reports on downlink bridges are not forwarded.
- When join requests are received without a leave, it is assumed that the set top box is watching both channel.
- MXK IGMP snooping supports existing Max Video Streams and Multicast Control List functionality.
- Using the IP on a bridge IP address when a join request is sent to the upstream multicast headend device.

Join requests

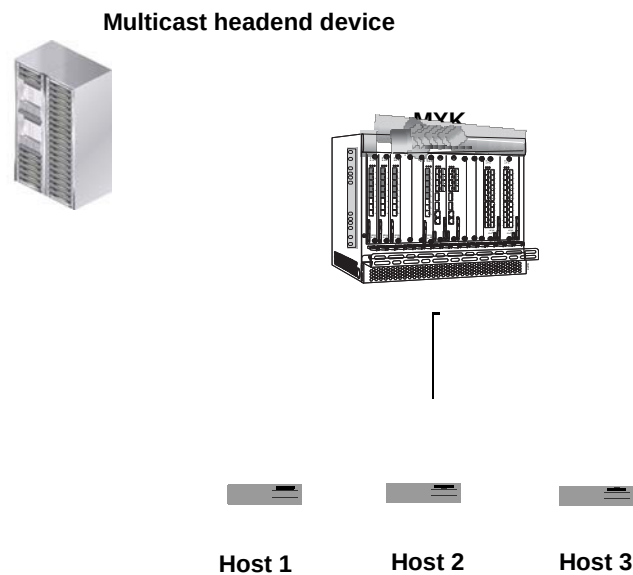
When you enable IGMP snooping, join requests from hosts are not forwarded by the MXK to the multicast headend device, but are tracked by the MXK in an information table where hosts are organized into a group. When a host sends a join request that is the first join request of the group, the MXK

terminates the join request from the host then originates a join request and sends it to the multicast headend device along with an IP address of 0.0.0.0 and a MAC address.



Note: The configured IP on a bridge IP address can be sent instead of 0.0.0.0. This provides the upstream multicast headend device the ability to distinguish between downstream MXKs for debugging purposes.

Figure 36: MXK and multicast head end device join and leave requests



Leave requests

When you enable IGMP snooping, leave requests from hosts are not forwarded by the MXK to the multicast headend device, but are tracked by the MXK in an information table where hosts are organized into a group. When a host sends a leave request that is the last leave request of the group, the MXK terminates the leave request from the host then originates a leave request and sends it to the multicast headend device. All leave requests, regardless of whether they are the last leave request of the group, or any earlier leave requests, are terminated on the MXK.

In this way, the multicast headend device starts and stops video transmission by processing requests sent directly from the MXK and not from downstream hosts.

IGMP snooping with proxy configuration

Enable IGMP snooping for video on uplink bridges when configuring the bridge path. The syntax is:

```
bridge-path add <interface/type> vlan <vlan-id> default igmpsnooping enable|
disable
```

The syntax to enable IGMP snooping, multicast aging, and IGMP query is:

```
bridge-path add <interface/type> vlan <vlan-id> default igmpsnooping enable
mcast <value> igmptimer <value>
```

The value for setting the *igmptimer* is in seconds.

Enabling IGMP snooping

To enable IGMP snooping with proxy, enter **bridge-path-add interface/type vlan vlan-id igmpsnooping enable**:

- 1 Create an uplink bridge on a 100/1000 Ethernet or a 10 GE port and designate a VLAN ID.

```
zSH> bridge add 1-a-4-0/eth uplink vlan 777
Adding bridge on 1-a-4-0/eth
Created bridge-interface-record ethernet4-777/bridge
```

The default for uplink bridges with VLAN IDs is tagged.

Verify the bridge.

```
zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
upl Tagged 777  ethernet4-777/bridge                DWN
```

View the **bridge-interface-record** profile.

```
zSH> get bridge-interface-record ethernet4-777/bridge
bridge-interface-record  ethernet4-777/bridge
vpi: -----> {0}
vci: -----> {0}
vlanId: -----> {777}
stripAndInsert: -----> {false} is passed up to the network
customARP: -----> {true}
filterBroadcast: -----> {true}
learnIp: -----> {false}
learnUnicast: -----> {false}
maxUnicast: -----> {0}
learnMulticast: -----> {false}
forwardToUnicast: -----> {true}
forwardToMulticast: -----> {true} passes video streams to interfaces that send a JOIN request
forwardToDefault: -----> {false}
bridgeIfCustomDHCP: -----> {true}
bridgeIfIngressPacketRuleGroupIndex: -> {0}
vlanIdCOS: -----> {0}
outgoingCOSOption: -----> {disable}
outgoingCOSValue: -----> {0}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
```

```

s-tagStripAndInsert: -----> {true}
s-tagOutgoingCOSOption: -----> {s-tagdisable}
s-tagIdCOS: -----> {0}
s-tagOutgoingCOSValue: -----> {0}
mcastControllist: -----> {}
maxVideoStreams: -----> {0}
isPPPoA: -----> {false}
floodUnknown: -----> {false}
floodMulticast: -----> {false}
bridgeIfEgressPacketRuleGroupIndex: --> {0}
bridgeIfTableBasedFilter: -----> {NONE(0)}
bridgeIfDhcpLearn: -----> {NONE(0)}
mvrVlan: -----> {0}

```

- 2 Add the bridge path and enable IGMP snooping. The default is disable.

```

zSH> bridge-path add ethernet4-777/bridge vlan 777 default igmpsnooping enable
Bridge-path added successfully

```

Verify the bridge path.

```

zSH> bridge-path show
VLAN/SLAN      Bridge                               Address
-----
777 ethernet4-777/bridge             Default, IGMP Proxy

```

Configuring IGMP snooping, multicast aging, and IGMP query

- 1 Create a tagged uplink bridge on a 100/1000 Ethernet or a 10 GE port with VLAN ID. Uplink bridges on the MXK default to tagged, even when *tagged* is not specified.

```

zSH> bridge add 1-a-5-0/eth uplink vlan 888
Adding bridge on 1-a-5-0/eth
Created bridge-interface-record ethernet5-888/bridge

```

Verify the bridge.

```

zSH> bridge show
Type VLAN      Bridge                               St  Table Data
-----
upl Tagged 777  ethernet4-777/bridge             DWN S VLAN 777 default
upl Tagged 888  ethernet5-888/bridge             UP

```

- 2 Add the bridge path and enable IGMP snooping and set the multicast aging period and the IGMP query interval.

The **mcast** sets the maximum age, in seconds, of a multicast packet before it is purged

The **igmp timer** indicates a time value in seconds. This value should be greater than 0. If you enter 0, the querying function is disabled.

```
zSH> bridge-path add ethernet5-888/bridge vlan 888 default igmpsnooping enable
igmptimer 120
Bridge-path added successfully
```

The *igmptimer* is now set for 120 seconds or two minutes.
Verify the bridge-path.

```
zSH> bridge-path show
VLAN/SLAN      Bridge                                     Address
-----
777 ethernet4-777/bridge                   Default, IGMP Proxy
888 ethernet5-888/bridge                   Default, IGMP Proxy
```

Creating a downlink bridge on an Active Ethernet port

You can create a downlink bridge on an Active Ethernet port with using a VLAN ID and an SLAN ID. You can also specify a maximum number of video streams and a multicast control list. Add the multicast control list and designate the maximum video streams using the *m/n* format. The multicast control list is set first and the maximum video streams second.

Members of the multicast control list must be defined to receive the video signal.

Create a downlink bridge on an Active Ethernet interface.

```
zSH> bridge add 1-13-1-0/eth downlink vlan 888 tagged video 1/3
Adding bridge on 1-13-1-0/eth
Created bridge-interface-record 1-13-1-0-eth-888/bridge
```

Verify the bridge.

```
zSH> bridge show
Type VLAN      Bridge                                     St  Table Data
-----
dwn Tagged 888  1-13-1-0-eth-888/bridge                   DWN
upl Tagged 777  ethernet4-777/bridge                   DWN S VLAN 777 default
upl Tagged 888  ethernet5-888/bridge                   UP  S VLAN 888 default
```

Enabling IGMP send IP address

You can configure the MXK to send the IP address used for IP on a bridge instead of 0.0.0.0 by entering **bridge-path-add <interface/type> vlan <vlan-id> igmpsnooping enable igmpsendip enable | disable** and set the **igmpsendip** parameter to *enable* with this command syntax.



Note: If you have not configured IP on a bridge on this MXK, a warning is displayed that there is no IP address to use.

- 1 Create the uplink bridge.

```
zSH> bridge add 1-a-8-0/eth uplink vlan 999
Adding bridge on 1-a-8-0/eth
Created bridge-interface-record ethernet8-999/bridge
```


- 2 Create the bridge path and enable *igmpsendip*.

The syntax is:

```
bridge-path add <interface/type> vlan <vlan-id> default igmpsnooping enable
igmpsendip enable igmptimer <value>
```

```
zSH> bridge-path add ethernet8-999/bridge vlan 999 default igmpsnooping enable
igmpsendip enable igmptimer 120
Bridge-path added successfully
```

Verify the bridge path.

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
dwn	Tagged 888	1-13-1-0-eth-888/bridge	DWN		
upl	Tagged 777	ethernet4-777/bridge	DWN	S VLAN 777	default
upl	Tagged 888	ethernet5-888/bridge	UP	S VLAN 888	default
upl	Tagged 999	ethernet8-999/bridge	DWN	S VLAN 999	default

To disable **igmpsendip**, enter *disable*.

To send the IP address of 0.0.0.0 enter **igmpsendip** *disable*.

Displaying bridge IGMP

The **bridge igmp** command displays the time left for multicast when entered from the card slot, not the MXK system.

In order to connect to a slot, the user entering the **bridge igmp** command *must* have *debug* permission.

- 1 To connect to the line card to view the time left on a multicast feed enter:

```
zSH> connect 6
```

- 2 To view IGMP data enter:

```
1/6-zSH> bridge igmp
```

Slan	Vlan	MAC Address	MCAST IP	Bridge Host MAC
0	999	01:00:5e:0a:0a:0a	224.10.10.10	1-6-1-0-eth
00:10:a4:c4:fe:95	02:14			
0	999	01:00:5e:7f:ff:fa	239.255.255.250	1-6-1-0-eth
00:10:a4:c4:fe:95	02:0			

In addition, you can run a **bridge igmp** command to determine whether IGMP is running on the system.

- 3 To exit from the line card enter:

```
1/6-zSH> exit
```



Note: It is important to exit from the line card as soon as you are finished viewing the IGMP data.

IGMPv3 messages respond for STBs

Some newer versions of Set Top Box (STB) support IGMPv2 and IGMPv3. This MXK release does not support IGMPv3 but it can respond to the IGMPv3 messages. When the MXK receives an IGMPv3 message, it will send out two general queries using IGMPv2. The STB will see these queries and will operate with IGMPv2 until the next reboot.

IGMP bridging statistics

Viewing IGMP bridge statistics



Note: The **ip igmpstat** command displays the ports receiving multicast traffic and the joined multicast group(s).

```
zSH> bridge igmpstat
```

		Received			Transmitted		
	Interface	GenQuery	SpecQuery	v2Report	Leave	GenQuery	SpecQuery
v2Report	Leave						
Processing list of 379							
	ethernet3-840	0	0	282735	0	0	0
0	0						
	ethernet3-101-502	0	0	0	0	0	0
0	0						
	ethernet3-500	85	8	20	89	0	0
0	0						
	ethernet3-0-503	0	0	0	0	0	0
0	0						
	ethernet3-121	0	0	0	0	0	0
0	0						
	ipobridge-94	0	0	0	0	0	0
0	0						
	ethernet3-848	0	0	0	0	0	0
0	0						
	ethernet3-94	0	0	7	7	0	0
0	0						
	ethernet3-2001	0	0	0	0	0	0
0	0						
	1-17-48-0-adsl-0-35	0	0	0	0	0	0
0	0						

IGMP bridging statistics

1-17-47-0-ads1-0-35	0	0	0	0	0	0
0 0						
1-17-46-0-ads1-0-35	0	0	0	0	0	0
0 0						
1-17-45-0-ads1-0-35	0	0	0	0	0	0
0 0						
1-17-44-0-ads1-0-35	0	0	0	0	0	0
0 0						

LINK AGGREGATION CONFIGURATION

This chapter describes MXK link aggregation on Ethernet cards and includes:

- [Link aggregation overview, page349](#)
- [Configure link aggregation on Ethernet uplink ports, page353](#)
- [Configure link aggregation on Ethernet line card ports, page357](#)
- [lacp command, page360](#)
- [Configure link aggregation bridges, page361](#)

Link aggregation overview

This section describes:

- [Link aggregation and LACP, page350](#)
- [Link aggregation modes, page350](#)
- [Link resiliency, page351](#)
- [Ethernet uplink ports available for link aggregation, page351](#)

The MXK supports 802.3ad link aggregation on the uplink and Ethernet line cards. Link aggregation allows aggregating physical 10 GE or 100/1000 Ethernet uplink ports into one single aggregated logical port for additional bandwidth capacity and resiliency. A link aggregation group can consist of up to eight ports.



Note: The 10 GE and the 100/1000 Ethernet ports cannot be aggregated into the same link aggregation group.



Note: Link aggregation is not supported across cards.

The MXK Ethernet cards that support link aggregation are:

- MXK-UPLINK-2X10G-8X1GE
- MXK-UPLINK-8X1GE
- MXK-UPLINK-4X1GE-CU
- MXK-UPLINK-4X1GE

- MXK-AE20-FE/GE-2S
- MXK-AE20-FE/GE

Link aggregation and LACP

The MXK uplink cards also support Link Aggregation Control Protocol (LACP), a layer 2 protocol used between network elements to exchange information regarding a link’s ability to be aggregated with other similar links.

For redundant configurations, the Ethernet ports on both the active uplink card and the redundant uplink must be configured for LACP. For redundant uplink card configurations, the link aggregated ports on each card provide redundant uplink port protection.



Note: You will need to configure the Ethernet switch on the remote end for link aggregation.

Link aggregation modes

Link aggregation has four modes with the default set to *on*:

- *on*
The setting to support manual link aggregation. The Ethernet link is aggregated manually with the **linkagg group** command.
LACP messages are not sent from this port, and LACP messages received on this port are ignored.
- *active*
The setting that supports LACP. Enables the Ethernet link to send and receive LACP messages and automatically link aggregates when the remote system responds with the appropriate LACP messages.
- *passive*
The setting that sets a link to receive LACP messages, and responds with LACP when receiving a far-end LACP initiation.
- *off*
The setting that turns all link aggregation functionality off. The Ethernet port cannot be aggregated either manually or dynamically.
LACP messages are not sent from this port, and LACP messages received on this port are ignored.

[Table10](#) shows the compatibility matrix for the four settings.

Table 10: LACP compatibility matrix settings

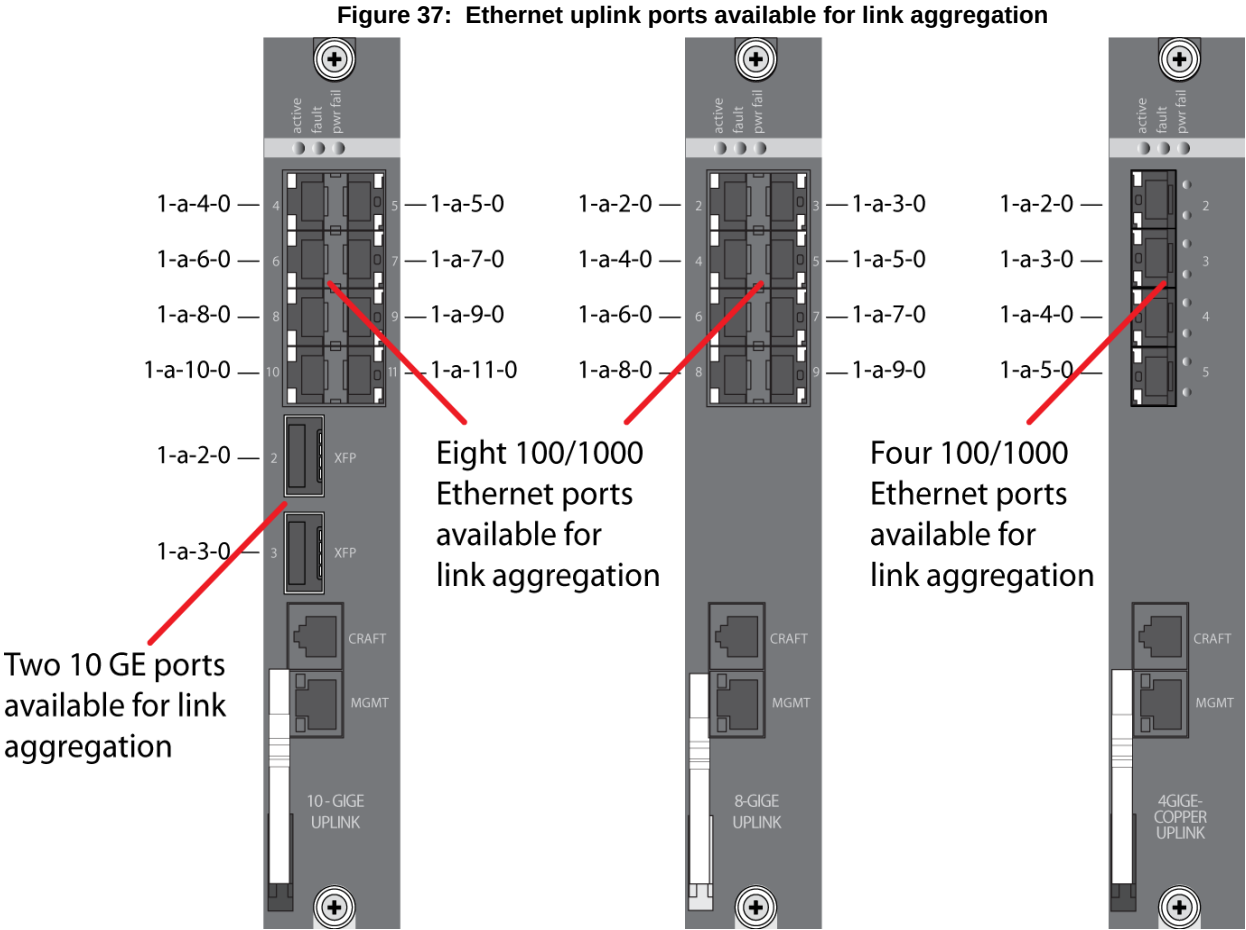
Device one	Device two	Comments
<i>active</i>	<i>active</i>	Both devices are sending and receiving LACP. Recommended setting for dynamic aggregation.
<i>active</i>	<i>passive</i>	One side of the connection between devices attempts to negotiate a aggregated group. Functional, but not recommended.
<i>on</i>	<i>on</i>	Only allows manual aggregation of links. Recommended if the far-end device is not capable of LACP.

Link resiliency

The link aggregation stays up as long as one link in the group is operational. Link aggregation manages links as they fail and come up again with no interruption in service. However, if all the links in a link aggregation group fail, the link aggregation group changes to a down state until a physical link is restored.

Ethernet uplink ports available for link aggregation

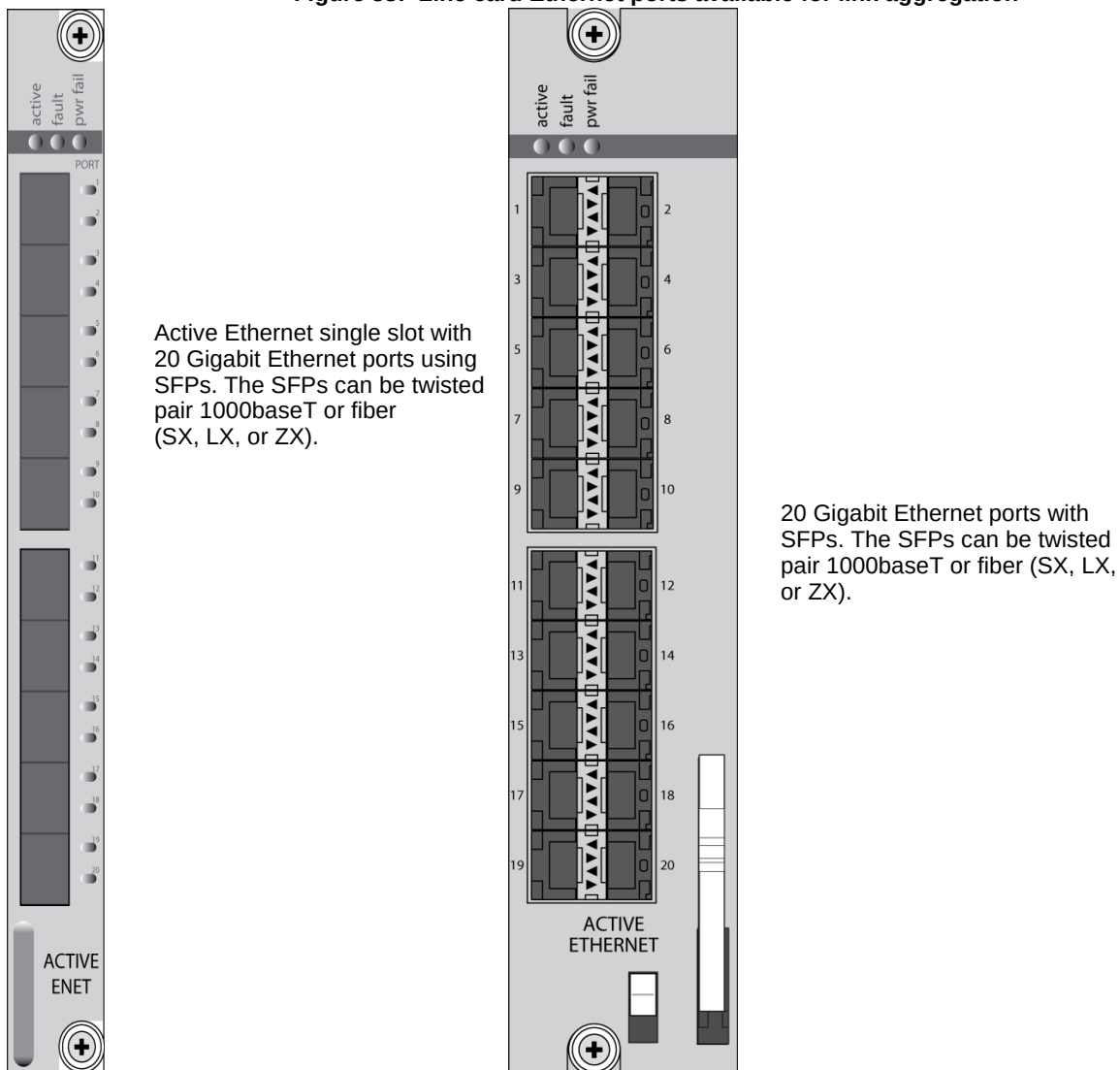
[Figure37](#) illustrates the physical ports available on the MXK uplink cards for link aggregation.



Ethernet line card ports available for link aggregation

Figure38 illustrates the physical ports available on the MXK Ethernet line cards for link aggregation.

Figure 38: Line card Ethernet ports available for link aggregation



Configure link aggregation on Ethernet uplink ports

This section discusses:

- [Configure Ethernet uplink ports for manual link aggregation, page354](#)
- [Configure Ethernet uplink ports for LACP, page355](#)
- [Delete a link aggregation group, page356](#)

Configuring the MXK to run link aggregation basically involves the choice of two modes, *on* and *active*.

When the mode is *on*, link aggregation groups are manually created by entering a group name and adding each link with the **linkagg add group** command from the CLI.

When the mode is *active*, the mode is changed from *on* to *active* with the **linkagg update link** *interface/type on | active* command and the link aggregation group is automatically created and is composed of up to two links depending on the remote device.

Configure Ethernet uplink ports for manual link aggregation

Link aggregation on the MXK, for either the eight 100/1000 Ethernet interfaces or two 10 GE interfaces, can be performed from the command line interface. The syntax for the **linkagg** command is:

zSH> **linkagg**

Usage: linkagg <add|delete> group <aggregation name/type> link <linkname/type>
|linkagg update link <linkname/type> <newvalue> |linkagg show

Creating link aggregated ports manually

To create a link aggregation between two 100/1000 Ethernet ports, for example, *1-a-7-0/eth* and *1-a-9-0/eth* use the **linkagg add group** command.

- 1 Make the physical connections between the two devices.
- 2 Assign a link, *1-a-7-0/eth*, to the link aggregation group *1-a-1-0/linkagg*.

zSH> **linkagg add group 1-a-1-0/linkagg link 1-a-7-0/eth**
Link aggregation successfully created.

group is a user-defined string.

Use the same **group** name for both of the ports you are aggregating.

- 3 Assign the second Ethernet port to the link aggregation group *1-a-1-0/linkagg*.

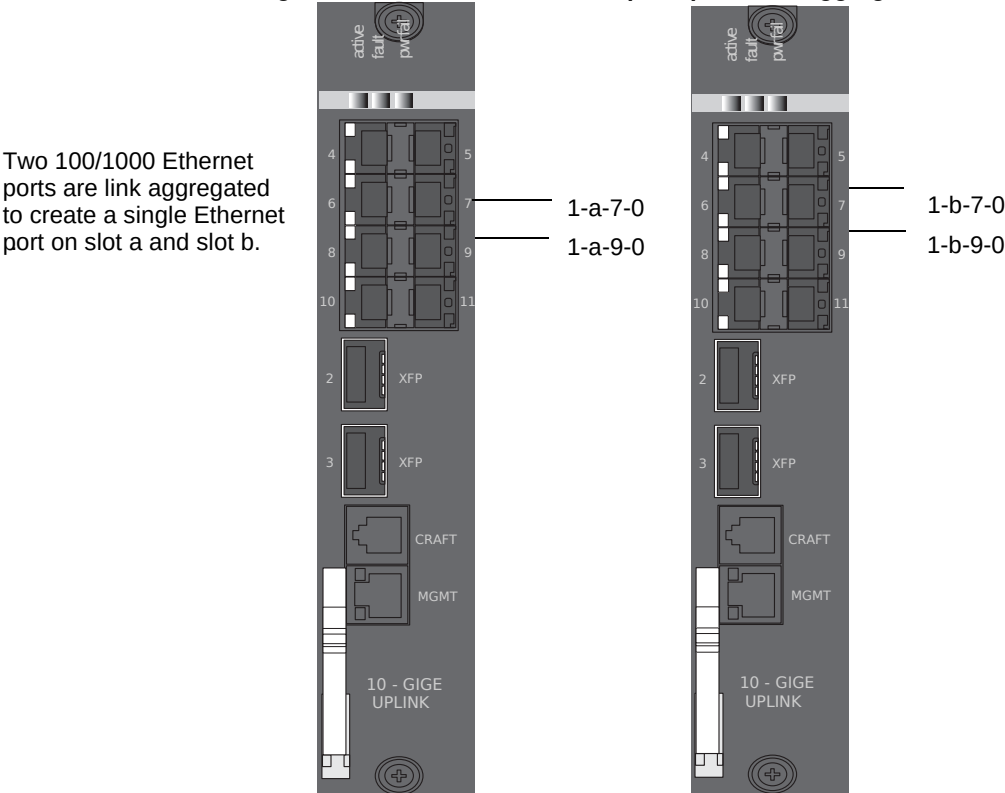
zSH> **linkagg add group 1-a-1-0/linkagg link 1-a-9-0/eth**
Link aggregation successfully created.



Note: For redundant uplink card configurations, the **linkagg add** automatically creates the corresponding redundant link in slot *b*.

[Figure39](#) describes the two 100/1000 Ethernet physical ports after they are aggregated to create a single Ethernet port.

Figure 39: Ethernet redundant uplink ports link aggregated



4 Enter the **linkagg show** command to view the ports just aggregated.

```
zSH> linkagg show
LinkAggregations:
slot unit ifName          partner: Sys          Pri    grp ID  admin
numLinks
-----
a    1    1-a-1-0          00:00:00:00:00:00    0x0    0x0    up      2
      links      slot    port    subport    admin
      1-a-9-0    a      9      0          up
      1-a-7-0    a      7      0          up
```

Configure Ethernet uplink ports for LACP

When the aggregation mode on the Ethernet uplink ports is *active*, the device sends and receives LACP and the link aggregation is dynamic, i.e., groups are created automatically. The mode is changed from *on* to *active* on Ethernet ports from the CLI with the **linkagg update link interface/type on | active** command.

Enabling LACP on Ethernet uplink ports

Enable two or more Ethernet ports on the uplink card for LACP.

1 Connect the MXK to the LACP enabled switch.

2 Change the mode of the links from *on* to *active*.

When a port on the uplink card is configured for LACP, the redundant card is automatically configured for LACP as well.

```
zSH> linkagg update link 1-a-10-0/eth active
```

Warning: this command will similarly update the aggregationMode of every link which is in an aggregation with this link, as well as any redundant peers.

Also, changing a link from on or off to active or passive will put the link into an aggregation if it is not in one.

Do you want to continue? [yes] or [no]: **yes**

```
zSH> linkagg update link 1-a-11-0/eth active
```

Warning: this command will similarly update the aggregationMode of every link which is in an aggregation with this link, as well as any redundant peers.

Also, changing a link from on or off to active or passive will put the link into an aggregation if it is not in one.

Do you want to continue? [yes] or [no]: **yes**

3 Verify the change from *on* to *active*.

```
zSH> get ether 1-a-11-0/eth
```

```
ether 1-a-11-0/eth
```

```
autonegstatus: ----> {enabled}
```

```
mauType: -----> {mau1000baseLxf}
```

```
restart: -----> {norestart}
```

```
ifType: -----> {mau1000baseLxf}
```

```
autonegcap: -----> {b10baseTFD+b100baseTXFD+b1000baseTFD}
```

```
remotefault: -----> {noerror}
```

```
clksrc: -----> {automatic}
```

```
pauseFlowControl: -> {disabled}
```

```
aggregationMode: --> {active} <-----
```

```
linkStateMirror: --> {0/0/0/0/0}
```

Delete a link aggregation group

Deleting the link aggregation group

Delete both LACP link aggregation groups and manually created link aggregation groups from the CLI with the **linkagg delete group interface/ type link interface/type** command.

Delete each link individually from the link aggregation group.

```
zSH> linkagg delete group 1-a-1-0/linkagg link 1-a-10-0/eth
```

Link successfully deleted from aggregation.

```
zSH> linkagg delete group 1-a-1-0/linkagg link 1-a-11-0/eth
```

Link successfully deleted from aggregation.



Note: If a *linkagg* bridge exists on the physical interface associated with the link aggregation group, you will not be able to delete the links.

Configure link aggregation on Ethernet line card ports

This section discusses:

- [Configure Ethernet line card ports for manual link aggregation, page357](#)
- [Configure line card Ethernet ports for LACP, page359](#)
- [Delete a link aggregation group, page360](#)

Configuring the MXK to run link aggregation basically involves the choice of two modes, *on* and *active*.

When the mode is *on*, link aggregation groups are created from the CLI by entering a group name and adding each link with the **linkagg add group** command.

When the mode is *active*, the mode is changed from *on* to *active* with the **linkagg update link interface/type on | active** command and the link aggregation group is automatically created and is composed of up to two links depending on the remote device.

Configure Ethernet line card ports for manual link aggregation

Link aggregation on the MXK can be performed from the CLI. All 20 Ethernet ports available on the single and the double-slot Ethernet cards are available for link aggregation.

The syntax for the **linkagg** command is:

```
zSH> linkagg
Usage: linkagg <add|delete> group <aggregation name/type> link <linkname/type> |
linkagg update link <linkname/type> <newvalue> |
linkagg show
```

Creating link aggregated ports manually

To create a link aggregation between two 10/100/1000 Ethernet ports, for example, *1-13-1-0/eth* and *1-13-2-0/eth*, use the **linkagg add group** command.

- 1 View the Ethernet cards available for link aggregation.

```
zSH> slots
Uplinks
a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
3: MXK 4 PORT GPON (RUNNING)
4: MXK 4 PORT GPON (RUNNING)
```

13: MXK 20 ACT ETH (RUNNING)

- 2 Create the link aggregation group by assigning a link to the group, in this case *1-13-1-0/linkagg*.

```
zSH> linkagg add group 1-13-1-0/linkagg link 1-13-1-0/eth
Interface 1-13-1-0/linkagg does not exist
Link aggregation successfully created.
```

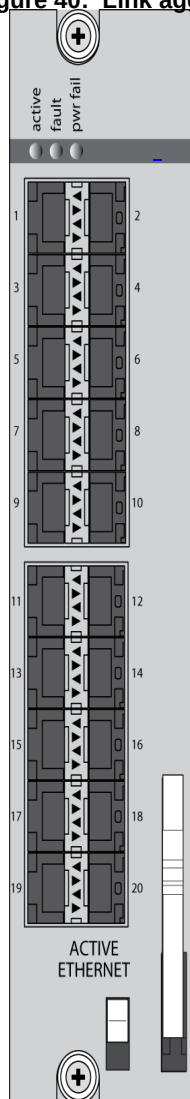
group is a user-defined string.

Use the same group name for the port you are aggregating.

- 3 Add another link to the link aggregation group.

```
zSH> linkagg add group 1-13-1-0/linkagg link 1-13-2-0/eth
Link aggregation successfully created.
```

Figure 40: Link aggregated line card Ethernet ports



1-13-1-0/eth

1-13-2-0/eth

Two Ethernet line card ports are aggregated into the *1-13-1-0/linkagg* group.

Figure40 describes the physical Ethernet ports after they are aggregated to create a single Ethernet port.

4 Verify the link aggregation group.

```
zSH> linkagg show
```

```
LinkAggregations:
```

slot	unit	ifName	partner: Sys	Pri	grp ID	admin	numLinks
13	1	1-13-1-0	00:00:00:00:00:00	0x0	0x0	up	2
		links	slot port subport		admin		
		1-13-1-0	13 1 0		up		
		1-13-2-0	13 2 0		up		

Configure line card Ethernet ports for LACP

When the aggregation mode on the Ethernet line card ports is *active*, the device sends and receives LACP and the link aggregation is dynamic, i.e., groups are created automatically. The mode is changed from *on* to *active* on from the CLI with the **linkagg update link interface/type on | active** command.

Enabling LACP on Ethernet line card ports

Enable two or more Ethernet ports on the line card for LACP.

- 1 Connect the MXK to the LACP enabled switch.
- 2 Change the mode of the links from *on* to *active*.

```
zSH> linkagg update link 1-13-1-0/eth active
```

```
Warning: this command will similarly update the aggregationMode of every link which
```

```
is in an aggregation with this link, as well as any redundant peers.
```

```
Also, changing a link from on or off to active or passive will put the link into an aggregation if it is not in one.
```

```
Do you want to continue? [yes] or [no]: yes
```

```
zSH> linkagg update link 1-13-2-0/eth active
```

```
Warning: this command will similarly update the aggregationMode of every link which
```

```
is in an aggregation with this link, as well as any redundant peers.
```

```
Also, changing a link from on or off to active or passive will put the link into an aggregation if it is not in one.
```

```
Do you want to continue? [yes] or [no]: yes
```

Delete a link aggregation group

Deleting the link aggregation group

Delete both LACP link aggregation groups and manually created link aggregation groups from the CLI with the **linkagg delete group** *interface/type link interface/type* command.

Delete each link individually from the link aggregation group.

```
zSH> linkagg delete group 1-13-1-0/linkagg link 1-13-1-0/eth
Link successfully deleted from aggregation.
```

```
zSH> linkagg delete group 1-13-1-0/linkagg link 1-13-2-0/eth
Link successfully deleted from aggregation.
```

lacp command

Use the **lacp** command to verify that the *aggregation partner key* number of the LACP enabled link aggregation group match, and view other link aggregation information.

lacp command syntax usage:

```
zSH> lacp
```

```
Usage: lacp <agg|id|monitor|state> [portNo] | lacp stats [portNo] [clear]
```

After connecting the MXK to an LACP enabled switch, you can verify that the *aggregation partner key* number matches for each link to the switch.

```
zSH> lacp monitor 2
```

```
PORT 2:
```

```
selected = SELECTED Disabled Traffic Enabled
```

```
actor state:7c
```

```
partner state:5a
```

```
1: partner key 4002 par port pri 8000, partner port # 2, actor state AGGREGATION  
SYNCHRONIZATION COLLECTING DISTRIBUTING DEFAULTED, partner state LACP_TIMEOUT  
SYNCHRONIZATION COLLECTING DEFAULTED
```

```
partner system: 00:00:00:00:00:00
```

```
1: agg id f03f5e0, par sys pri: 8000, agg partner key 4002
```

```
par sys: 00:00:00:00:00:00
```

```
zSH> lacp monitor 3
```

```
PORT 3:
```

```
selected = SELECTED Disabled Traffic Enabled
```

```
actor state:7c
```

```
partner state:5a
```

```
2: partner key 4002, par port pri 8000, partner port # 3, actor state  
AGGREGATION SYNCHRONIZATION COLLECTING DISTRIBUTING DEFAULTED , partner state  
LACP _TIMEOUT SYNCHRONIZATION COLLECTING DEFAULTED
```

```
partner system: 00:00:00:00:00:00
```

```
2: agg id f03f500, par sys pri: 8000, agg partner key 4002
```

```
par sys: 00:00:00:00:00:00
```


Configure link aggregation bridges

Bridge interfaces can be added to link aggregation ports for bridging.

Configure link aggregation uplink bridges

Creating link aggregated uplink bridges

Unlearned traffic received on this interface is forwarded to the external network.

- 1 To verify link aggregation groups, enter:

```
zSH> linkagg show
LinkAggregations:
slot unit ifName          partner: Sys          Pri    grp ID  admin  numLinks
-----
a  1  1-a-1-0
    links                00:00:00:00:00:00    0x0    0x0    up    1
    slot      port      subport      admin
-----
    1-a-3-0          a      3      0          up
b  2  1-b-2-0            00:00:00:00:00:00    0x0    0x0    up    1
    links                slot      port      subport      admin
-----
    1-b-8-0          b      8      0          up
a  2  1-a-2-0            00:00:00:00:00:00    0x0    0x0    up    1
    links                slot      port      subport      admin
-----
    1-a-8-0          a      8      0          up
b  1  1-b-1-0            00:00:00:00:00:00    0x0    0x0    up    1
    links                slot      port      subport      admin
-----
    1-b-3-0          b      3      0          up
```

- 2 To create an uplink bridge with link aggregation, enter:

```
zSH> bridge add 1-a-1-0/linkagg uplink vlan 333 tagged
Adding bridge on 1-a-1-0/linkagg
Created bridge-interface-record linkagg-a-1-333/bridge
```

- 3 Add the bridge path.

```
zSH> bridge-path add linkagg-a-1-333/bridge vlan 333 default
Bridge-path added successfully
```

- 4 To delete a bridge with link aggregation enter:

```
zSH> bridge-path delete linkagg-a-1-333/bridge vlan 333 default
Delete complete

zSH> bridge delete linkagg-a-1-333/bridge vlan 333
linkagg-a-1-333/bridge Delete complete
```

Creating an uplink bridge on a aggregated Ethernet port

If a bridge is created on a link aggregated Ethernet interface on a physical port, a linkagg bridge is automatically created.

Create the uplink bridge.

```
zSH> bridge add 1-a-2-0/eth uplink vlan 777
Adding bridge on 1-a-2-0/eth
Created bridge-interface-record linkagg-a-1-777/bridge
```

Since the Ethernet port *1-a-2-0/eth* is part of a link aggregation group, the bridge type is automatically designated *linkagg*.

Verify the *linkagg* bridge.

```
zSH> bridge show
Type VLAN          Bridge                               St  Table Data
-----
upl Tagged 777    linkagg-a-1-777/bridge             DWN S VLAN 777 default
```

Configure link aggregation line card bridges

Creating a link aggregated bridge on an Ethernet line card

1 Verify the link aggregation group.

```
zSH> linkagg show
LinkAggregations:
slot unit ifName          partner: Sys          Pri    grp ID  admin  numLinks
-----
  13   1   1-13-1-0          00:00:00:00:00:00    0x0    0x0    up     2
      links          slot    port    subport          admin
      -----
      1-13-1-0        13      1      0              up
      1-13-2-0        13      2      0              up
```

2 Create the bridge on the link aggregation group.

```
zSH> bridge add 1-13-1-0/linkagg downlink vlan 600
Adding bridge on 1-13-1-0/linkagg
Created bridge-interface-record linkagg-13-1/bridge
```

3 View the bridge created on the link aggregation group.

```
zSH> bridge show
Type VLAN          Bridge                               St  Table Data
-----
upl Tagged 200    ethernet5-200/bridge             UP  S VLAN 200 default
upl Tagged 999    ethernet5-999/bridge             UP  S VLAN 999 default
dwn              600    linkagg-13-1/bridge             DWN
3 bridges displayed
```

Deleting a link aggregation bridge

Delete the link aggregation bridge.

```
zSH> bridge delete linkagg-13-1/bridge vlan 600  
linkagg-13-1/bridge delete complete
```


8

MXK 100/1000 ETHERNET AND 10 GE UPLINK CARDS

This chapter describes the MXK 100/1000 Ethernet and 10 GE uplink cards and uplink card configuration:

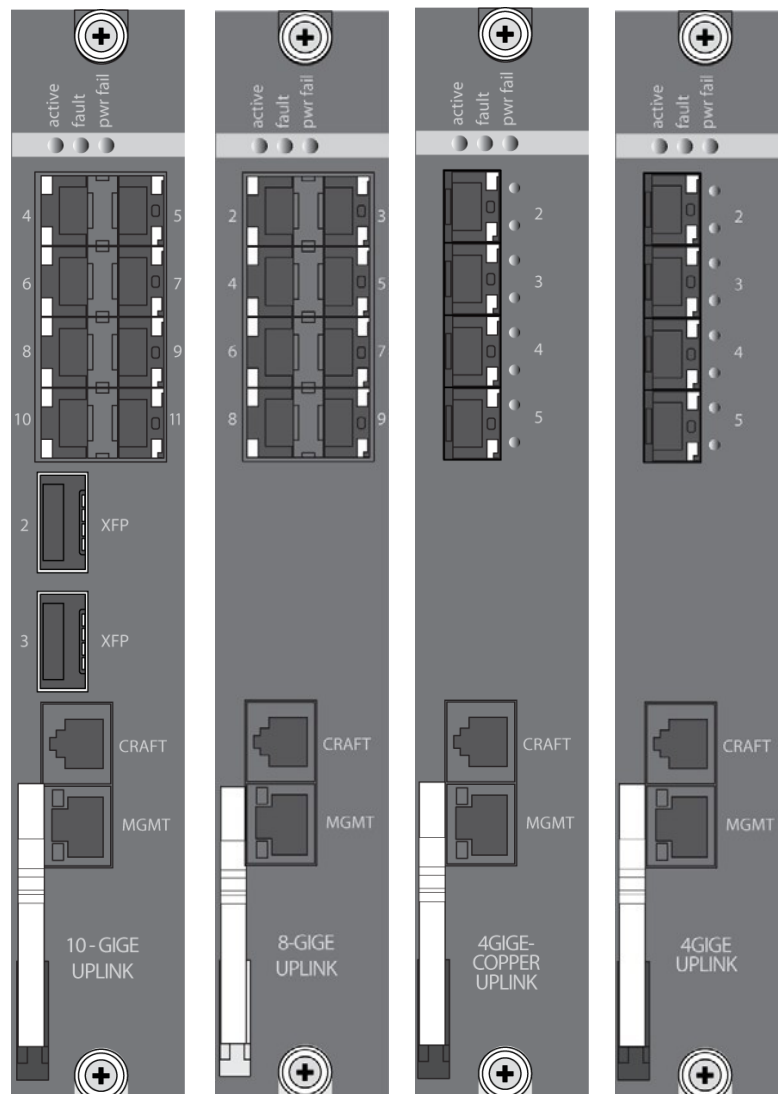
- [100/1000 Ethernet and 10 GE uplink cards, page366](#)
- [EAPS, page376](#)
- [Small form factor pluggables, page383](#)
- [Uplink card pinouts, page383](#)

100/1000 Ethernet and 10 GE uplink cards

This section describes the MXK 100/1000 Ethernet and 10 GE uplink cards and how to configure the cards.

- [100/1000 Ethernet and 10 GE uplink card overview, page366](#)
- [Uplink card specifications, page367](#)
- [Redundant uplink card configuration, page369](#)
- [Disable Tx power on the uplink standby card, page372](#)
- [View additional card and system information, page373](#)
- [Displaying and updating Ethernet interfaces, page374](#)

100/1000 Ethernet and 10 GE uplink card overview



The MXK uplink cards are the MXK-UPLINK-2X10G-8X1GE, the MXK-UPLINK-8X1GE, MXK-UPLINK-4X1GE-CU, and the MXK-UPLINK-4X1GE.

The MXK MXK-UPLINK-2X10G-8X1GE uplink card provides high-speed Gigabit Ethernet interfaces with active/standby redundancy. This uplink card consists of two 10 GE and eight 100/1000 Ethernet interfaces.

The MXK MXK-UPLINK-8X1GE uplink card provides high-speed Gigabit Ethernet interfaces with active/standby redundancy. This uplink card consists of eight 100/1000 Ethernet interfaces.

The MXK-UPLINK-4X1GE-CU uplink card provides high-speed Gigabit Ethernet interfaces with active/standby redundancy. This uplink card consists of four 100/1000 Ethernet interfaces and supports only copper line cards.

The MXK-UPLINK-4X1GE uplink card provides high-speed Gigabit Ethernet interfaces with active/standby redundancy. This uplink card consists of four 100/1000 Ethernet interfaces and supports all line cards.

All the uplink cards provide a serial craft interface and a 10/100 Ethernet interface for management. The 100/1000 and 10 GE ports are available for link aggregation.

The 100/1000 Ethernet and 10 GE interfaces support a number of small form factor pluggables (SFPs) and XFPs respectively that enable the card to interface with a variety of media types. (For more information see [Small form factor pluggables on page383](#).)

Uplink card specifications

[Table11](#) provides the MXK-UPLINK-2X10G-8X1GE uplink card specifications.

Table 11: Uplink card MXK-UPLINK-2X10G-8X1GE specifications

Specification	Description
Size	1 slot
Physical interfaces	Two 10 GE ports with XFPs. See Chapter 15, Small Form Factor Pluggable (SFP) Connectors, on page757. Eight 100/1000 Ethernet ports with SFPs. The SFPs can be twisted pair 1000baseT or fiber (SX, LX or ZX). See Small form factor pluggables on page383. The optical interfaces are class 1 Laser International Safety Standard IEC 825 compliant RJ45 Ethernet 10/100 Ethernet interface for management RS232D serial craft interface

Table 11: Uplink card MXK-UPLINK-2X10G-8X1GE specifications (Continued)

Specification	Description
Standards supported	AF-PHY-0086.001 G.965 and ETSI EN 300 347-1 V2.2.2 (V5.2) Gigabit Ethernet (GigE) IEEE 802.3
Management interface	RS-232D serial craft port Management Ethernet 10/100 port routable for out-of-band management. SNMP—None or all.
Redundancy	Card redundancy
Power consumption	88 W

[Table12](#) provides the MXK-UPLINK-8X1GE uplink card specifications.

Table 12: Uplink card MXK-UPLINK-8X1GE specifications

Specification	Description
Size	1 slot
Physical interfaces	Eight 100/1000 Ethernet ports with SFPs. The SFPs can be twisted pair 1000baseT or fiber (SX, LX or ZX). See Small form factor pluggables on page383 . The optical interfaces are class 1 Laser International Safety Standard IEC 825 compliant RJ45 Ethernet 10/100 Ethernet interface for management RS232D serial craft interface
Standards supported	AF-PHY-0086.001 G.965 and ETSI EN 300 347-1 V2.2.2 (V5.2) Gigabit Ethernet (GigE) IEEE 802.3
Management interface	RS-232D serial craft port Management Ethernet 10/100 port routable for out-of-band management. SNMP—None or all.
Redundancy	Card redundancy
Power consumption	80 W

[Table13](#) provides the MXK-UPLINK-4X1GE-CU uplink card specifications.

Table 13: Uplink card MXK-UPLINK-4X1GE-CU specifications

Specification	Description
Size	1 slot
Physical interfaces	Four 100/1000 Ethernet ports with SFPs. The SFPs are copper and fiber 100M/1G). See Small form factor pluggables on page383 . The optical interfaces are class 1 Laser International Safety Standard IEC 825 compliant RJ45 Ethernet 10/100 Ethernet interface for management RS232D serial craft interface
Standards supported	AF-PHY-0086.001 G.965 and ETSI EN 300 347-1 V2.2.2 (V5.2) Gigabit Ethernet (GigE) IEEE 802.3
Management interface	RS-232D serial craft port Management Ethernet 10/100 port routable for out-of-band management. SNMP–None or all.
Redundancy	Card redundancy
Power consumption	50W

Redundant uplink card configuration

This section describes how to configure a redundant uplink card. After insertion, the uplink card in slot *a* automatically configures when the software image is on the flash.



Note: MXK uplink cards must be installed in the middle slots a and b of the MXK chassis.

[Table14](#) provides the type and software image for the uplink cards on the MXK. The parameters for the software image of the card type are found in the **card-profile** for that software image.

Table 14: MXK uplink card types

Card	Type	Name of software image
MXK-UPLINK-2X10G-8X1GE	10100	<i>mxup2tg8g.bin</i>
MXK-UPLINK-8X1GE	10101	<i>mxupg8g.bin</i>
MXK-UPLINK-4X1GE-CU	10121	<i>mxup4gcopper.bin</i>
MXK-UPLINK-4X1GE	10125	<i>mxup4g.bin</i>

Caution:

You must configure redundant physical interfaces on *both* the active and standby cards. In addition, you must manually keep the configuration of the physical interfaces on the active and standby cards in sync.

Each uplink card must run the same software version and have the same size flash card.



Note: When configuring the redundant uplink card, the settings in the **card-profile** for the both cards must be identical.

Configuring redundant uplink cards

To configure redundant uplink cards on the MXK:

- 1 Insert the active uplink card in slot *a*.
- 2 Insert the redundant uplink card in slot *b*.

The group ID of the **card-profile** for slot *b* must match the group ID of the **card-profile** of slot *a*.

To modify a parameter in the **card-profile**, use the **update card-profile shelf/slot/card** type command.

- 3 Verify the state of the uplink card in slot *b*.

```
zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (NOT_PROV)
Cards
  4: MXK 4 PORT GPON (RUNNING)
  13: MXK 20 ACT ETH (RUNNING)
```

- 4 Verify the group ID of the uplink card in slot *a*.

```
zSH> get card-profile 1/a/10100
card-profile 1/a/10100
sw-file-name: -----> {mxup2tg8g.bin}
admin-status: -----> {operational}
upgrade-sw-file-name: ---> {}
upgrade-vers: -----> {}
admin-status-enable: ----> {enable}
sw-upgrade-admin: -----> {reloadcurrrev}
sw-enable: -----> {true}
sw-upgrade-enable: -----> {false}
card-group-id: -----> {1} group ID
hold-active: -----> {false}
weight: -----> {nopreference}
card-line-type: -----> {unknown-type}
card-atm-configuration: -> {notapplicable}
card-line-voltage: -----> {not-used}
```

```

maxvpi-maxvci: -----> {notapplicable}
card-init-string: -----> {}
wetting-current: -----> {disabled}

```

- 5 Change the group ID in the **card-profile** of slot *b* by entering the **card add** command specifying slot *b* and *group*.

```

zSH> card add b group 1
new card-profile 1/b/10100 added, sw-file-name "mxup2tg8g.bin", 1 option:
card-group-id 1

```

The system updates the **sw-file-name** to match the **card-profile** of slot *a* and changes the **card-group-id** parameter to *1*.

- 6 Verify the **sw-file-name** and **card-group-id** parameters in the **card-profile** of the stand-by uplink card in slot *b*.

```

zSH> get card-profile 1/b/10100
card-profile 1/b/10100
sw-file-name: -----> {mxup2tg8g.bin}
admin-status: -----> {operational}
upgrade-sw-file-name: ---> {}
upgrade-vers: -----> {}
admin-status-enable: ----> {enable}
sw-upgrade-admin: -----> {reloadcurrrev}
sw-enable: -----> {true}
sw-upgrade-enable: -----> {false}
card-group-id: -----> {1}
hold-active: -----> {false}
weight: -----> {nopreference}
card-line-type: -----> {unknown-type}
card-atm-configuration: -> {notapplicable}
card-line-voltage: -----> {not-used}
maxvpi-maxvci: -----> {notapplicable}
card-init-string: -----> {}
wetting-current: -----> {disabled}

```

- 7 To view the status of both uplink cards enter **slots**:

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  4: MXK 4 PORT GPON (RUNNING)
  13: MXK 20 ACT ETH (RUNNING)

```

- 8 To view card information including the state of the card and how long the card has been running, enter **slots** and specify the slot number of the card:

```

zSH> slots a
Type          : *MXK TWO TENGIGE EIGHT GIGE
Card Version  : 800-02485-01-A
EEPROM Version: 1
Serial #      : 1769060

```

```

CLEI Code       : No CLEI
Card-Profile ID : 1/a/10100
Shelf           : 1
Slot            : a
ROM Version     : MALC CAN MXK-20-Feb-2007
Software Version: MXK 1.16.1.128
State           : RUNNING
Mode            : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat   : 80
Fault reset     : enabled
Uptime          : 19 days, 23 hours, 50 minutes

```

```

zSH> slots b
Type           : MXK TWO TENGIGE EIGHT GIGE
Card Version    : 800-02485-01-A
EEPROM Version  : 1
Serial #        : 1568340
CLEI Code       : No CLEI
Card-Profile ID : 1/b/10100
Shelf           : 1
Slot            : b
ROM Version     : MALC CAN MXK-20-Feb-2007
Software Version: MXK 1.16.1.128
State           : RUNNING
Mode            : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat   : 79
Fault reset     : enabled
Uptime          : 19 days, 23 hours, 58 minutes

```

Disable Tx power on the uplink standby card

The **line-red set** command can be used to disable the laser on the redundant uplink card when using a fiber optic SFP used to configure Virtual Router Redundancy Protocol (VRRP).



Note: After a fail over, there will be additional latency to bring the laser back up.

Disabling the Tx power on the uplink standby card

- 1 Disable the laser on the MXK standby card.

```
zSH> line-red set 1-a-2-0/eth standbytx disable
```

- 2 View the *standbytx* status.

zSH> **line-red show 1-a-2-0/eth**

redundancy status for 1-a-2-0/eth:

		NOREBOOT	standbytx	DISABLE	timeout 0	NONREVERTIVE	revert	timeout 0
Interface-Type	Interface-Name	Oper-State	Oper-Status					
=====	=====	=====	=====					
Primary	1-a-2-0/eth	Active	UP					
Secondary	1-b-2-0/eth	Standby	OOS					

View additional card and system information

View the EPROM version of the card:

zSH> **eeshow card a**

EEPROM contents: for slot 30

EEPROM_ID : 00 -- CARD

Version : 01

Size : 054

CardType : 10100 -- MXUP2TG8G

CardVersion : 800-02485-01-A

SerialNum : 01769060

ShelfNumber : 00001

CLEI Code : No CLEI

Cksum : 0x8596

MAC Address are enabled for 11 address(es) starting at:

00:01:47:1A:FE:64

View the ROM version of the card:

zSH> **romversion a**

MXK 1.14.3.100

May 20 2008, 21:44:05

Verify the existence of a daughter card on the system:

zSH> **eeshow 1d**

EEPROM contents: for slot 30

EEPROM_ID : 01 -- 1DAUGHTER

Version : 01

Size : 022

CardType : 10100 -- MXUP2TG8G

CardVersion : 800-02482-01-B

SerialNum : 01768971

ShelfNumber : 00001

CLEI Code : No CLEI

Cksum : 0xFA1F

Verify the existence of a second daughter card by entering **eeshow 2d**.

View the version of the system software:

zSH> **swversion**

Zhone mxUp2Tg8g software version MXK 1.16.1.128

Displaying and updating Ethernet interfaces

The **list**, **get**, and **update** commands support use of the interface *shelf-slot-port-subport/eth* syntax to facilitate Ethernet port and interface monitoring and configuration.

To list the currently configured Ethernet interfaces, enter the **list ether** command.

```
zSH> list ether
ether 1-a-1-0/eth
ether 1-a-2-0/eth
ether 1-a-3-0/eth
ether 1-a-4-0/eth
ether 1-a-5-0/eth
ether 1-a-6-0/eth
ether 1-a-7-0/eth
ether 1-a-8-0/eth
ether 1-a-9-0/eth
ether 1-a-10-0/eth
ether 1-a-11-0/eth
ether 1-b-1-0/eth
ether 1-b-2-0/eth
ether 1-b-3-0/eth
ether 1-b-4-0/eth
ether 1-b-5-0/eth
ether 1-b-6-0/eth
ether 1-b-7-0/eth
ether 1-b-8-0/eth
ether 1-b-9-0/eth
ether 1-b-10-0/eth
ether 1-b-11-0/eth
ether 1-13-1-0/eth
ether 1-13-2-0/eth
ether 1-13-3-0/eth
ether 1-13-4-0/eth
...
42 entries found.
```

The **list ether** command shows the Ethernet interfaces on each uplink card, in slot a and slot b, as well as the Ethernet interfaces on the Active Ethernet card.

The **slots** command verifies the location of the cards with Ethernet interfaces:

```
zSH> slots
Uplinks
a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
4: MXK 4 PORT GPON (RUNNING)
13: MXK 20 ACT ETH (RUNNING)
```

To view an Ethernet interface, enter the **get ether** command followed by the interface index.

```
zSH> get ether 1-13-4-0/eth
ether 1-13-4-0/eth
autonegstatus: ----> {enabled}
mauType: -----> {mau1000basetfd}
restart: -----> {norestart}
ifType: -----> {mau1000basetfd}
autonegcap: -----> {b100baseTX+b100baseTXFD+b1000baseT+b1000baseTFD}
remotefault: -----> {noerror}
clksrc: -----> {automatic}
pauseFlowControl: -> {disabled}
aggregationMode: --> {off}
linkStateMirror: --> {0/0/0/0/0}
```

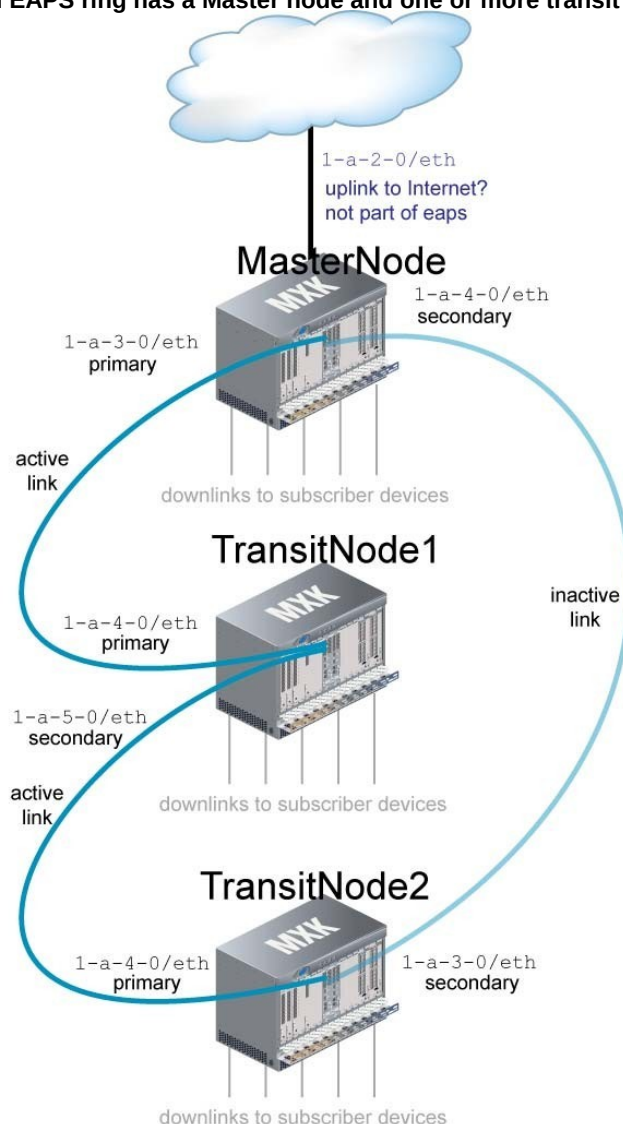
EAPS

The Ethernet Automatic Protection Switching (EAPS) protocol (RFC3619) creates a fault tolerant system of links by providing fast switchover from a primary path to a secondary path for each VLAN.

In many access and Metro Area Network (MAN) topologies, transport between nodes, and from nodes to the distribution layer of the network is required to run over a physical ring infrastructure. In these cases, multi-platform support for a packet ring protocol is needed to connect and communicate over the ring topology.

An EAPS Domain exists on a single Ethernet ring. Any Ethernet Virtual Local Area Network (VLAN) that is to be protected is configured on all ports in the ring for the given EAPS Domain. Each EAPS domain has a single “master node.” All other nodes on that ring are referred to as “transit nodes.”

Figure 41: An EAPS ring has a Master node and one or more transit nodes



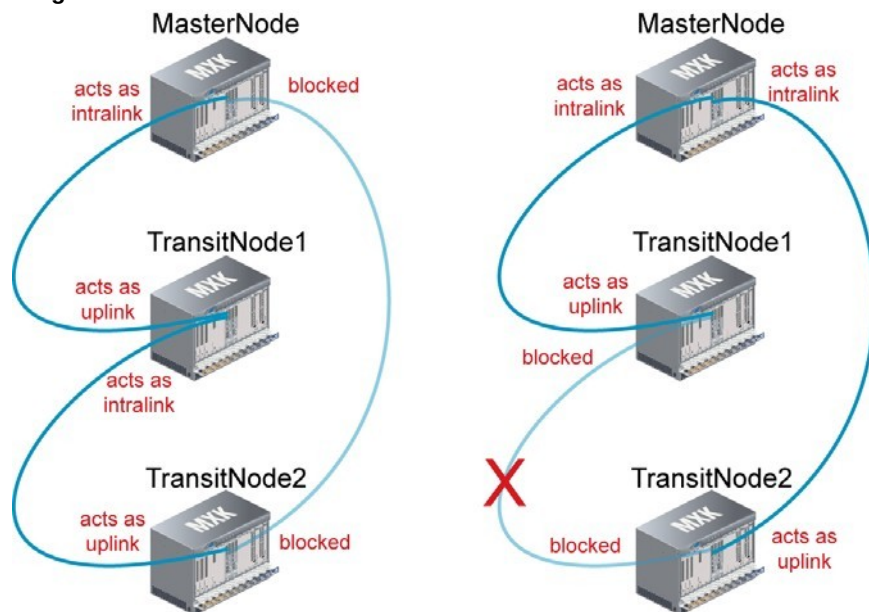
EAPS, which is only for layer 2 bridging, takes advantage of multiple VLANs being on the same physical link. One VLAN is a control VLAN which sends Health Check messages around the ring on a control ring. Because a bridge cannot loop back on itself, there is always a blocked link for data on the EAPS ring, but not for the control ring, so the control messages can loop around, but data does not.

Figure 42: With multiple VLANs on a link one is a control VLAN with protects the other VLAN



Each node on the ring has two ports which are part of the ring. One is the primary port and one secondary. On a master node the primary port (in its initial state) will act as an intralink, the other port will be block. For the transit nodes one rlink (the name for these bridge interfaces on the transit node) will act as an uplink. If the other node is going to another transit node, the other rlink will act as an intralink. On the last link which loops back to the master node, the rlink will be blocked.

Figure 43: When a Link Down condition is detected the protected VLANs change



If a transit node detects that a link has gone down, it sends a Link Fault message to the master. So any time there is a delayed Health Check message

or a Link Fault message the master will change the direction messages go to the other nodes.

Because a bridge cannot loop back on itself, there is always a blocked link for data on the same physical link as the control node, so the control messages can loop around, but data does not.

When a link is down EAPS messages on the control VLAN notify the other nodes which essentially clear their bridging data bases and switch how each link will act. For a master node, the blocked node will function as an intralink. The other link on the master node and rlinks adjust as well depending on which link the break occurs.

To create an EAPs ring

The following steps create an EAPS ring as shown in [Figure41 on page 376](#).

1 Configure the master node

- a** Configure the node as a master node, define the **domain_id**, define the primary and secondary interfaces and define the control VLAN

```
zSH> eaps add domain MasterNode master 1-a-3-0/eth 1-a-4-0/eth control
100
```

- b** Add the protected VLANs to the domain

```
zSH> eaps-vlan add domain MasterNode 200-300,999
```

- c** Enable the node

```
zSH> eaps enable domain MasterNode
```

- d** Add the bridges to the node

```
zSH> bridge add 1-a-2-0/eth uplink vlan 200 tagged (upstream to cloud, not part of
eaps)
zSH> bridge add 1-a-3-0/eth downlink vlan 200 tagged (primary link)
zSH> bridge add 1-a-4-0/eth downlink vlan 200 tagged (secondary link)
zSH> bridge-path add ethernet2-200/bridge vlan 200 default
```

2 Configure the first transit node

- a** Configure the node as a transit node, define the **domain_id**, define the primary and secondary interfaces and define the control VLAN

```
zSH> eaps add domain TransitNode1 transit 1-a-4-0/eth 1-a-5-0/eth
control 100
```

- b** Add the protected VLANs to the domain

```
zSH> eaps-vlan add domain TransitNode1 200-300,999
```

- c** Enable the node

```
zSH> eaps enable domain TransitNode1
```

d Add the bridges to the node

```

zSH> bridge add 1-a-4-0/eth rlink vlan 200 tagged
zSH> bridge add 1-a-5-0/eth rlink vlan 200 tagged
zSH> bridge-path add ethernet4-200 bridge vlan 200 rlink
zSH> bridge-path add ethernet5-200 bridge vlan 200 rlink

```

3 Configure the second transit node**a** Configure the node as a transit node, define the **domain_id**, define the primary and secondary interfaces and define the control VLAN

```

zSH> eaps add domain TransitNode2 transit 1-a-4-0/eth 1-a-5-0/eth
control 100

```

b Add the protected VLANs to the domain

```

zSH> eaps-vlan add domain TransitNode2 200-300,999

```

c Enable the node

```

zSH> eaps enable domain TransitNode2

```

d Add the bridges to the node

```

zSH> bridge add 1-a-4-0/eth rlink vlan 200 tagged
zSH> bridge add 1-a-3-0/eth rlink vlan 200 tagged
zSH> bridge-path add ethernet4-200 bridge vlan 200 rlink
zSH> bridge-path add ethernet3-200 bridge vlan 200 rlink

```

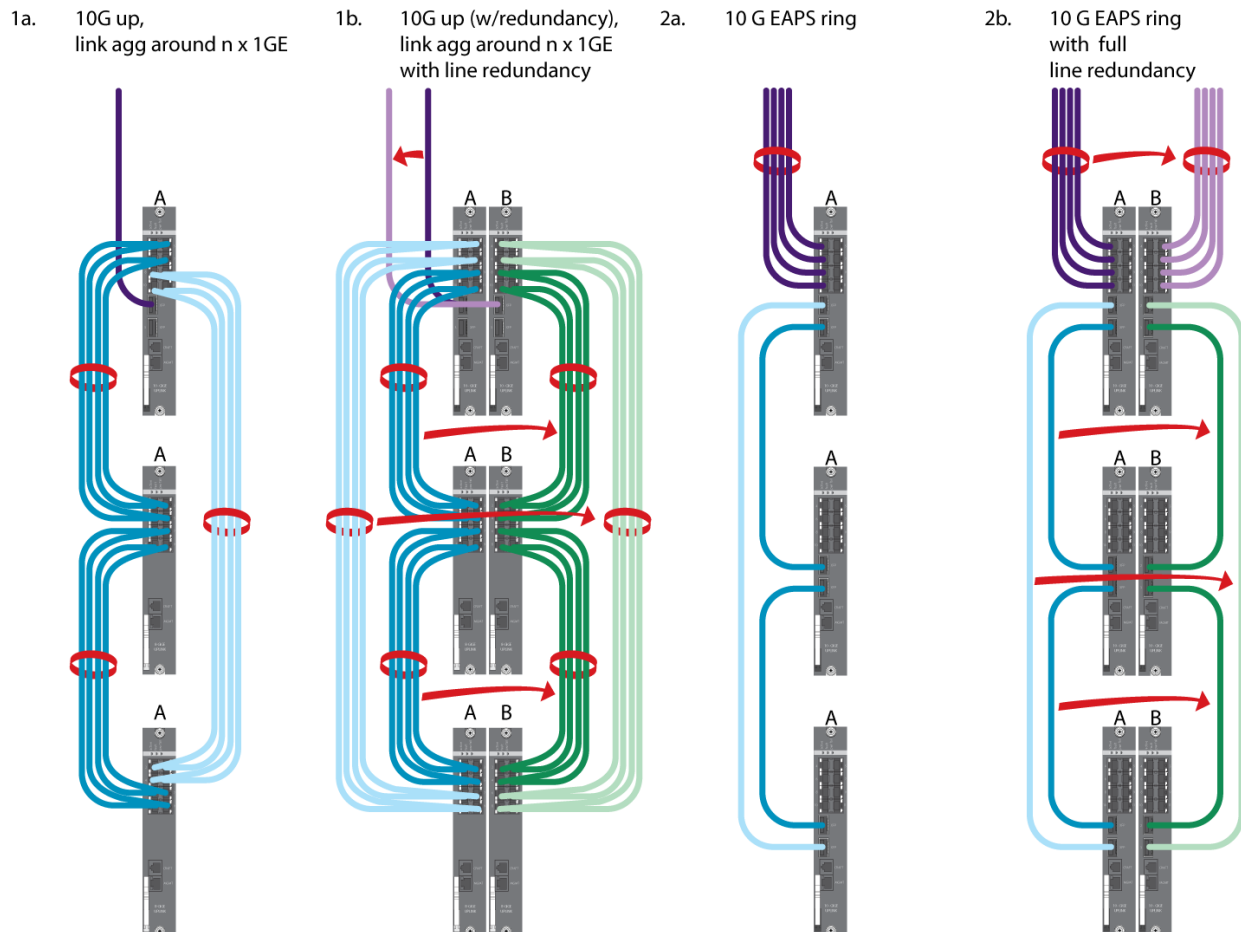
Advanced EAPs topics

As shown in [Figure42 on page 377](#), a single physical link may have many EAPS domain, each domain with a control VLAN and protected VLANs. A node may also be a master node for one EAPS domain and a transit node for another.

Common EAPs scenarios

For the MXK there are two basic common scenarios, having a 10Gig uplink with the EAPS ring being aggregated on the 1Gig Ethernet interfaces, or having the EAPS ring on the 10Gig interfaces and the uplink on aggregated 1 GigE links. Redundancy on the uplinks and EAPs rings would expand these basic scenarios.

Figure 44: The most common EAPS scenarios include aggregating links or using 10G interfaces to create the EAPS ring



eaps

The **eaps** command configures an EAPS master node or transit node.

Syntax **eaps** <add|delete|disable|enable|help|modify|show>

Syntax **eaps add domain** *domain_id* <master|transit> control *vlan*
<interface1> *interface1/type* <interface2> *interface2/type*
<interval value> <drop value> <timeout value> <trap
[yes|no]> <cntlpri value> <protpri value>

The **eaps add** command creates a node as an EAPS master or transit node, defines which interfaces are primary and secondary interfaces for the node, defines the control VLAN for the EAPS node (the control VLAN must be the same as the other nodes in the EAPS ring), and defines other variables such as the time between Health Check messages, the amount of time before determining a link fault, whether to send SNMP alert messages and defining the priority of messages on the control and protected VLANs. A newly added node is placed in the Inactive state

domain *domain_id*

domain is the alpha-numerical value (i.e. a string name) of the domain. It is user assigned and should represent the string description of the Domain's purpose (for example, which ring the node belongs to as in "South_Campus", which node, "transit_node_2", or a combination "South_Campus_transit_node_2").

master|transit

The **master|transit** parameter defines the state of the node on a ring; only one node for a given domain and control VLAN can be master, all other nodes are required to be transit nodes.

<interface1> *interface1/type*

The **interface1** parameter defines the first port or Link Aggregation group; if a node is configured as Master, then **interface1** will be considered the primary port. The string token "interface1" is optional on the **eaps add** command, but mandatory for the **eaps modify** command

<interface2> *interface2/type*

The **interface2** parameter defines the first port or Link Aggregation group; if a node is configured as Master, then **interface2** will be considered the primary port. The string token "interface2" is optional on the **eaps add** command, but mandatory for the **eaps modify** command

control *vlan*

The **control** parameter denotes the control VLAN (outermost VLAN for Q-in-Q non-802.1Q support) for the EAPS ring Domain. All nodes on the same EAPS ring must use the same control VLAN.

interval *time*

The interval parameter sets the time between Health messages (frequency) to be transmitted out of the primary ring port. The interval parameter is only valid for master nodes.

timeout *time*

The **timeout** parameter sets the timeout duration. For a Master Node this is the time since the last receipt of a HELLO message before the master node declares a ring fault. For a Transit Node this is the time the Node will remain in state Preforwarding before it transitions (in the event of a dropped a Ring Up message) to the Health state.

trap <on | off>

The trap parameter determines whether the node will send an SNMP alert describing the state change.

cntlpri priority

The **cntlpri** parameter sets the priority that Control messages will be sent.

protpri priority

The **protpri** parameter sets the priority that messages on protected VLAN will be sent.

Syntax **eaps modify domain** *domain_id* <master|transit> <control vlan> <interface1 interface1/type> <interface2 interface2/type> <interval value> <drop value> <timeout value> <trap [yes|no]> <cntlpri value> <protpri value>

The **eaps modify** command changes the configured variables of and EAPS node. The EAPS node must be disabled before using the **eaps modify** command.

Syntax **eaps show domain** <*domain_id*|all>

The **eaps show** command displays the protected VLAN list for the domain if the list exists. The command returns an error if the supplied **domain_id** does not exist.

domain domain_id|all>

domain is the alpha-numerical value (i.e. a string name) of the domain. It is user assigned and should represent the string description of the Domain's purpose (for example, which ring the node belongs to as in "South_Campus", which node, "transit_node_2", or a combination "South_Campus_transit_node_2").

all displays all eaps connections on the node.

Syntax **eaps disable domain** *domain_id*

The **eaps disable** command disables an existing domain on the node. (The node no longer moves traffic. To modify a node using the **eaps modify** command, the node must first be disabled.)

Syntax **eaps enable domain** *domain_id*

The **eaps enable** command enables an existing domain on the node. The **eaps enable** command moves the domain from a disabled state to an enabled state, so the node may pass messages on protected VLANs.

eaps-vlan

The **eaps-vlan** command creates, modifies, deletes or displays protected VLANs on an EAPS node.

Syntax

Syntax **eaps-vlan** <add | delete | modify | show > domain *domain_id* *VLAN_List*

Syntax `eaps-vlan add domain domain_id VLAN_List`

The **eaps-vlan add** command creates a Protected VLAN or a list of protected VLANs for an existing domain. The command will return an error if the supplied *domain_id* does not already exist.

Protected VLAN lists consist of comma separated entries, an entry can be either a single VLAN Id, or a dash separated range of ascending VLAN Ids. VLAN Ids must be greater than zero and less than 4090. You may use at most four entries. No spaces are permitted. e.g. "eaps-vlan add domain metro2 100-113,205,370-420,999" would create a list of protected VLANs for Domain "metro2" consisting of VLANs 100 through 113, VLAN 205, VLANs 370 through 420 and VLAN 999.

Syntax `eaps-vlan delete domain domain_id VLAN_List`

The **eaps-vlan delete** command removes a Protected VLAN or a list of protected VLANs for an existing domain. The command will return an error if the supplied *domain_id* does not already exist.

Syntax `eaps-vlan modify domain domain_id VLAN_List`

The **eaps-vlan modify** command alters a Protected VLAN or a list of protected VLANs for an existing domain. The command will return an error if the supplied *domain_id* does not already exist.

Syntax `eaps-vlan show domain <domain_id|all>`

The **eaps-vlan modify** command displays the list of protected VLANs for an existing domain. The command will return an error if the supplied *domain_id* does not already exist.

Small form factor pluggables

Zhone Technologies supports a variety of small form factor pluggables (SFPs) and XFPs which are selected depending on the protocol, fiber type and distance requirements. For information and specifications on supported SFPs and XFPs, see [Chapter 15, Small Form Factor Pluggable \(SFP\) Connectors](#), on page 757.

Uplink card pinouts

This section lists the pinouts for the following interfaces that are common on all the uplink cards:

- [Serial \(craft\) port pinouts](#), page 384
- [Ethernet port pinouts](#), page 384

For information about other port pinouts for uplink cards, refer to the chapters for each type of card, later in this manual.

Serial (craft) port pinouts

[Table15](#) lists the uplink cards' serial (craft) port pinouts. The serial (craft) port is an RS232 D type configured as DTE.

Table 15: Uplink card serial (craft) port pinouts

Pin	Function
1	DCE Ready, Ring Indicator (DSR/RI)
2	Received Line Signal Detector (DCD)
3	DTE Ready (DTR)
4	Signal Ground (SGND)
5	Received Data (RD)
6	Transmitted Data (TD)
7	Clear To Send (CTS)
8	Request To Send (RTS)

[Table16](#) lists the pinouts to connect a DB9 connector to the MXK RJ45 serial craft port.

Table 16: RJ45 to DB9 adapter pinouts

RJ-45 pin	Color	Function	DB-9 pin
1	N/A	DCE Ready, Ring Indicator (DSR/RI)	not used
2	N/A	Received Line Signal Detector (DCD)	not used
3	N/A	DTE Ready (DTR)	not used
4	Red	Signal Ground (SGND)	5
5	Green	Received Data (RD)	2
6	Yellow	Transmitted Data (TD)	3
7	N/A	Clear To Send (CTS)	Looped to pin 8
8	N/A	Request To Send (RTS)	Looped to pin 7

Ethernet port pinouts

[Table17](#) lists the Ethernet port pinouts on the uplink cards.

Table 17: Uplink card Ethernet port pinouts

Pin	Function
1Tx	+
2Tx	-
3Rx	+
4Not	used
5Not	used
6Rx	-
7Not	used
8Not	used

9 MXK GPON CARDS

This chapter describes the MXK Gigabit Passive Optical Networks (GPON) cards and GPON card configuration:

- [GPON cards, page388](#)
- [GPON configuration, page392](#)
- [Manage ONU with OMCI, page434](#)
- [Dynamic GEM ports, page451](#)
- [Bandwidth allocation, page459](#)
- [OMCI Statistics, page471](#)
- [PON Statistics, page474](#)

GPON technology provides one of the most cost effective ways for service providers to deploy fiber based services to the residential subscribers, businesses or other types of node. Utilizing GPON splitters which can be co-located with the MXK or remotely in the optical deployment network (ODN), the service provider can determine the best topology for their network.

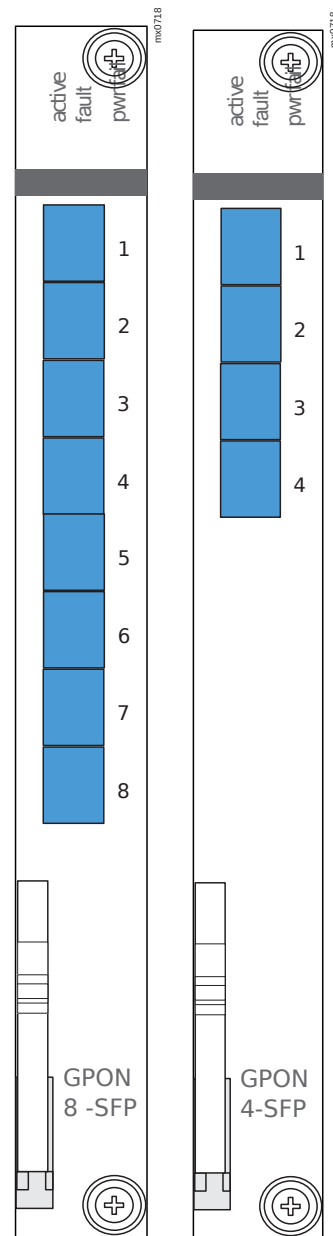
Zhone's line of zNID GPON Residential Gateways complete Zhone's GPON solution with video, voice and data support.

GPON cards

This section describes the MXK GPON cards and how to configure the cards.

- [GPON card overview, page388](#)
- [GPON card specifications, page389](#)
- [GPON card configuration, page390](#)
- [View additional card and system information, page391](#)

GPON card overview



Zhone provides two GPON line cards, the MXK-GPONX8-IO and the MXK-GPONX4-IO. Both GPON cards support 2.5 Gbps downstream bandwidth and 1.25 Gbps upstream bandwidth per interface as specified in the G.984.1-4 specifications.

The MXK-GPONX8-IO line card has an octal-port interface that provides industry leading capabilities. The MXK 8 port GPON card can support up to 512 GPON subscribers.

The MXK-GPONX4-IO line card has a quad-port interface. The MXK 4 port GPON card can support up to 256 GPON subscriber.

The SFPs used in the MXK GPON cards are the MXK-GPON-SFP-B+ and MXK-GPON-SFP-B+-RSSI. The MXK-GPON-SFP-B+-RSSI supports Received Signal Strength Indication (RSSI) feature. For more information about RSSI, please see [Received Signal Strength Indication \(RSSI\) and Digital Diagnostic Monitoring \(DDM\), page431](#).

AES encryption of 128 bits is supported on the GPON OLT chipset.

See [Chapter 4, IP Configuration](#), [Chapter 5, Bridging Configuration](#), and [Chapter 6, Video Configuration](#) for procedures on configuring routes, bridges, and video.

The following features are supported:

- Class B+ Optics with -28dB link budget
- 64 subscribers per OLT interface
- RF Overlay (1550nm wavelength)
- GPON type B redundancy
- Traffic management for IP QoS, traffic shaping
- RSSI support (available for MXK-GPON-SFP-B+-RSSI)

GPON card specifications

[Table 18](#) provides the MXK-GPONX8-IO and the MXK-GPONX4-IO card specifications.

Table 18: GPON OLT card specifications

Specification	Value
Size	1 slot
Density	8 port – 512 subscribers (64 subscribers per interface) 4 port – 256 subscribers (64 subscribers per interface)
Physical interfaces	SC-UPC fiber optic connector.
Line characteristics	Receives VoIP and data traffic at a 1310nm wavelength Transmits VoIP and data traffic at a 1490nm wavelength
Nominal line rate	2.5 Gbps downstream 1.25 Gbps upstream
Protocol support	Multicast IGMP v1/v2 Host-based routing Network-based routing IP host and gateway support DHCP server (RFC 2131, 2132), DHCP relay Bridging 802.1D VLAN 802.1Q/p Dense/sparse multicast
Power consumption	63 W (8 port) 50 W (4 port)

GPON card configuration

Each card installed in the system must have a **card-profile**. The line card type determines the parameter settings in the **card-profile** and the software image for the card. Performing a **card add <slot #>** automatically creates the **card-profile** for the card with the correct software image and settings.

[Table 19](#) provides the type and software image for the GPON cards on the MXK.

Table 19: MXK GPON card types

Card	Type	Name of software image
MXK-GPONX8-IO	10203	<i>mxlc8gp.bin</i>
MXK-GPONX4-IO	10205	<i>mxlc4gp.bin</i>

Creating a card-profile for a GPON card

Each card installed in the system must have a **card-profile**. The **card-profile** is created when you perform a **card add**.

- 1 Install a GPON card in the desired line card slot.
- 2 Configure a GPON card on the MXK:

```
zSH> card add 5
```

The system creates a **card-profile** for the GPON card in *slot 5*.

Verifying the line card installation

- 1 After adding a card to the MXK, you can verify the card by entering **slots**:

```
zSH> slots
```

Uplinks

a: *MXK EIGHT GIGE (RUNNING)

b: MXK EIGHT GIGE (RUNNING)

Cards

1: MXK 8 PORT GPON (RUNNING)

5: MXK 4 PORT GPON (RUNNING)

- 2 Enter the **slots** command and specify the slot number of the card to view card information and the state of the card. For example:

```
zSH> slots 5
```

Type : MXK 4 PORT GPON

Card Version : 800-02586-01-A

EEPROM Version : 1

Serial # : 1963593

CLEI Code : No CLEI

Card-Profile ID : 1/5/10205

Shelf : 1

```

Slot          : 5
ROM Version   : MXK 1.15.1.108
Software Version: MXK 1.16.2.119
State         : RUNNING
Mode          : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat  : 79
Fault reset    : enabled
Uptime        : 32 days, 19 hours, 14 minutes

```

- 3 View the **card-profile** of the card by entering **get card-profile shelf/slot/ cardtype**:

```

zSH> get card-profile 1/5/10205
card-profile 1/5/10205
sw-file-name: -----> {mxlc4gp.bin}
admin-status: -----> {operational}
upgrade-sw-file-name: ---> {}
upgrade-vers: -----> {}
admin-status-enable: ----> {enable}
sw-upgrade-admin: -----> {reloadcurrrev}
sw-enable: -----> {true}
sw-upgrade-enable: -----> {false}
card-group-id: -----> {0}
hold-active: -----> {false}
weight: -----> {nopreference}
card-line-type: -----> {unknowntype}
card-atm-configuration: -> {notapplicable}
card-line-voltage: -----> {not-used}
maxvpi-maxvci: -----> {notapplicable}
card-init-string: -----> {}
wetting-current: -----> {disabled}
pwe-timing-mode: -----> {none}

```

View additional card and system information

View the EPROM version of the card:

```

zSH> eeshow card 5
EEPROM contents: for slot 5
EEPROM_ID    : 00 -- CARD
Version      : 01
Size         : 054
CardType     : 10205 -- MXLC4GP
CardVersion  : 800-02586-01-A
SerialNum    : 01963593
ShelfNumber  : 00001
CLEI Code    : No CLEI
Cksum        : 0x23E6

```

View the version of the system software:

```

zSH> swversion
Zhone mxUp2Tg8g software version MXK 1.16.2.119

```

GPON configuration

This section includes the following topics:

- [Smart OMCI, page392](#)
- [GPON type B redundancy, page417](#)
- [MXK GPON using the Reg ID for provisioning, page424](#)
- [GPON ONU serial number format \(Hexadecimal or Decimal\), page426](#)
- [GPON extended reach, page428](#)
- [Received Signal Strength Indication \(RSSI\) and Digital Diagnostic Monitoring \(DDM\), page431](#)

Smart OMCI

This section includes the following topics:

- [OMCI overview, page392](#)
- [Smart OMCI overview, page393](#)
- [Configure Smart OMCI on ONU, page395](#)
- [Delete OMCI profile, page410](#)
- [Import and export OMCI profile, page413](#)

OMCI overview

The ONT Management and Control Interface (OMCI) is a protocol defined by ITU984.4 that enables the Optical Line Terminal (OLT) to control an Optical Network Unit (ONU). This protocol allows the OLT to:

- Establish and release connections across the ONU.
- Manage the User Network Interfaces (UNIs) at the ONU.
- Request configuration information and performance statistics.
- Autonomously inform the system operator of events such as link failures.

The OMCI protocol runs across the GEM connection between the OLT controller and the ONU controller that is established at ONU initialization. The OMCI protocol is asymmetric: the controller in the OLT is the master and the one in the ONU is the slave. A single OLT controller using multiple instances of the protocol over separate control channels may control multiple ONUs.

The ONU management and control interface requirements given in the ITU984.4. Recommendation are needed to manage ONU in the following areas:

- Configuration management
- Fault management

- Performance management
- Security management

Smart OMCI overview

The OMCI standard defines the Managed Entity (ME) commands that allow an OLT to communicate with an ONU. The ME commands are classified into two categories: Mandatory and Optional. The sequence in which the ME commands are sent from the OLT to the ONU is not always specified by the standard leading to vendor specific implementations of OMCI. With OMCI, service providers can encounter problems with OLT and ONU interoperability. In addition to vendor interoperability, there can also be interoperability of software versions. A new ONU model or software version can require an OLT software upgrade. Upgrading OLT software can cause the inconvenience and expense of a service outage for existing customers.

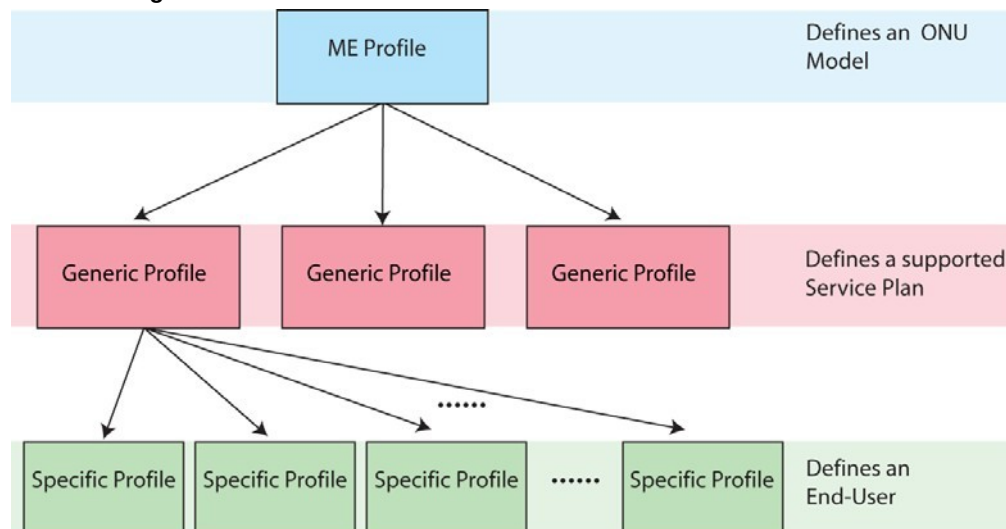
Zhone's Smart OMCI functionality, on the SLMS software product line, provides the flexibility to add new ONU models without upgrading the software on the OLT.

Implementing Smart OMCI

Smart OMCI functionality is implemented on the MXK by using OMCI profiles.

The three types of OMCI profiles defined in the system are ME, Generic, and Specific. Each profile type is synonymous to a task performed in the network deployment phase. As shown in the [Figure45](#), these three profile types have a hierarchical relationship.

Figure 45: Smart OMCI Architecture



- ME profile

The ME profile defines an ONU model.

The ME profile contains all the information required to support an ONU and defines the OMCI commands that OLT uses to configure an ONU. If a service provider supports 3 different ONUs in their network, there will be 3 ME profiles in the MXK. The ME profile is created on the MXK by an ME profile file that is downloaded from Zhone's website.

- Generic profile

The Generic profile defines the service plan supported by the service provider for a given ONU model.

A Generic profile is always associated with only one ME profile and contains the values for network parameters that define a service plan and the value for infrastructure network elements such as the softswitch IP address. If the service provider supports 5 different service plans on each of the 3 supported ONU models, there will be a total of 15 Generic Profiles in the MXK (5 Generic profiles for each of the ME profile). The Generic Profile can be created using the CLI, ZMS or WebUI. The ME profile and Generic profile are created at the time of initial network deployment before activating the user.

- Specific profile

The Specific profile defines the end-user and is created before activating the end-user. The Specific profile is always associated with only one Generic profile. The Specific profile contains value for variables and the variable list in the Specific profile is same as in the Generic profile. At creation, the Specific profile automatically inherits all the values of the parent Generic profile and does not require modification when the same values are used. When there is user specific information, like a telephone number, the values can be overridden by modifying those variables in the Specific profile. The variables defined in the Generic and Specific profiles are values used by the OMCI commands in the ME profile.

When activating an end-user, based on the service plan and ONU model being used by the end-user, choose the appropriate ME profile and Generic profile to associate with the Specific profile.

The order of precedence for implementing a value in the information field that is sent to the ONU is first the Specific profile, then the Generic profile, and finally the ME profile.

ME profiles and Generic profiles are normally created by a network analyst or network architect. The ME profile is the profile of the capabilities of the ONU model. Multiple MEs may be used for a single model. The more common strategy is to have all attributes for the ONU model configured in the ME profile. The Generic profile is intended to define ISP user bundles. If the ME profile has all ports configured, the Generic profile may define which are active for the end user. The specific profile is the end user profile and contains end user specific information, such as the phone number.

Configure Smart OMCI on ONU

Generally these are the steps to follow to configure the MXK to be able to manage ONU with Smart OMCI:

- [Creating ME profile file, page395](#)
- [Downloading ME profile file to MXK, page399](#)
- [Creating ME profile, page400](#)
- [Creating Generic profiles, page400](#)
- [Creating Specific profiles, page403](#)
- [Viewing current variable settings on an ONU, page405](#)
- [Activating an ONU with OMCI profiles, page407](#)

Creating ME profile file

Zhone Technologies provides the service provider a Smart OMCI web-interface to select desired ONU model and services.

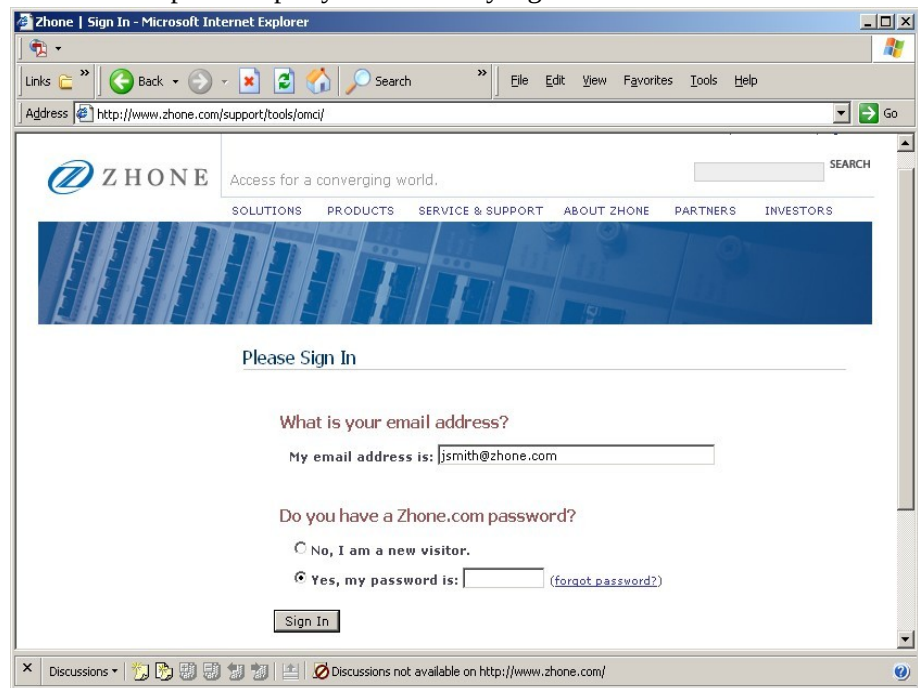
Using the Smart OMCI web-interface the service provider creates the ME profile file that containing the ME structure information which is unique to the ONU hardware model.

Access the Smart OMCI web-interface can be through Zhone's website.

To create an ME profile file:

- 1** Navigate to the Zhone website at "<http://www.zhone.com/support/tools/omci/>".
- 2** Access the website by entering the email address and the password selected at registration.

Note: skip this step if you are already signed in.



3 Select desired ONU model, then click **Continue**.

This example selects ONU model ZNID-GPON-2510.



- 4** After selecting the ONU model, the Smart OMCI web-interface updates to display the list of services that are supported on this ONU hardware model.
- 5** Select the desired services. For each service, you can select the supported physical interfaces, GEM Index, and VLAN filtering.

GEM index is in the range of 5xx to 35xx.

This example selects GEM index 5xx for data service on port *eth1* and *eth2*, GEM index 7xx for voice service on port *POTS1* and *POTS2*, GEM index 9xx for video service on port *eth3* and *eth4*.



Note: Take a note of the GEM index you selected for different services. It could be used to calculate the GEM port ID with the following formula:

GEM port ID = GEM index + ONU ID

The GEM port ID is used when you provisioning services on bridges or routers by using the **bridge add** or **host add** commands.

Refer to [Create a GEM port with a GPON traffic profile, page451](#) to get detail configuration.

ZNID-GPON-2510: Indoor GPON ONT, 4xFE, 2xPOTS

Zhone's ZNID-GPON-2510 is an indoor ONT designed for high performance applications. It features 4 10/100Mbps LAN Ports and 2 POTS Voice FXS ports in a compact enclosure for indoor use. 2510 may be installed as a desktop or, wall mounted and is powered via an AC Adaptor. 2510 ONT is ideal for triple-play service deployments in Fiber-to-the-Home/Premise application. Industry standard SIP

■ [ZNID-GPON-2510 Datasheet](#)

Data & Video Configuration

☒ **Port ETH 1** GEM: 5xx Data Add single tag to untagged frame
VLAN ID: 100 COS: 0 (802.1p)

☒ **Port ETH 2** GEM: 5xx Data Add single tag to untagged frame
VLAN ID: 100 COS: 0 (802.1p)

☒ **Port ETH 3** GEM: 9xx Video Add single tag to untagged frame
VLAN ID: 999 COS: 4 (802.1p)

☒ **Port ETH 4** GEM: 9xx Video -choose data service-

Voice Configuration

☒ **Port POTS 1**

☒ **Port POTS 2**

GEM: 7xx VLAN ID: 200 COS: 7 (802.1p)
Country Code: US
Host IP Option: Static
Netmask: 0.0.0.0
Gateway: 0.0.0.0
Protocol: SIP
VOIP Server: 0.0.0.0
Primary DNS: 0.0.0.0
Secondary DNS: 0.0.0.0
SIP Domain:

- 6 Click the **Create Configuration File** button. An ME profile file is created and displayed in the ME profile file page.

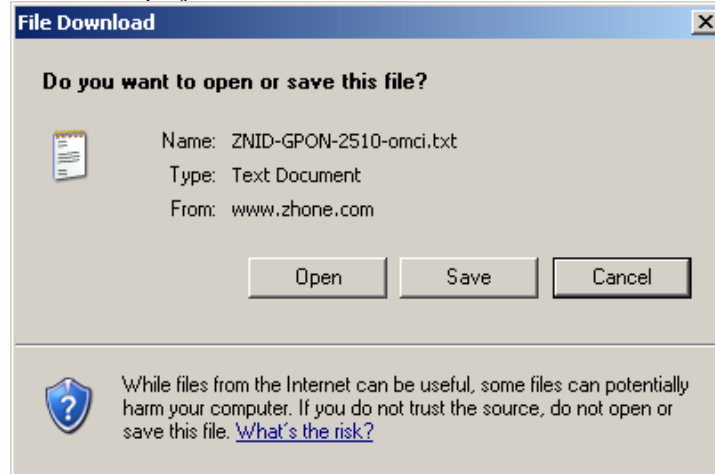
```

1  [ZONE_ME_PROFILE]
2  # OMCI configuration file for ZNID-GPON-2510
3  # Cambridge ONT with 4 Eth and 2 Pots
4  # Generated by csun@zhone.com at Thu Jul  1 11:27:05 2010
5  # Engine $Revision: 1.17 $ Tmpl $Revision: 1.9 $ Model $Revision: 1.29 $
6  #
7  # Selected configuration:
8  # eth1 (ETH 1): data over GEM port 5xx
9  # eth2 (ETH 2): data over GEM port 5xx
10 # eth3 (ETH 3): video, data over GEM port 9xx
11 # eth4 (ETH 4): video, data over GEM port 9xx
12 # pots1 (POTS 1): voice over GEM port 7xx using SIP
13 # pots2 (POTS 2): voice over GEM port 7xx using SIP
14 #
15
16 #=====
17 [Variable Definitions]
18 #=====
19
20 $autoDetectConfigEth1("ETH1 Auto Detection",string(32))
21 $vlEth1V1("ETH 1 Data VLAN 1 (VID or COS,VID)",string(32))
22 $autoDetectConfigEth2("ETH2 Auto Detection",string(32))
23 $vlEth2V1("ETH 2 Data VLAN 1 (VID or COS,VID)",string(32))
24 $autoDetectConfigEth3("ETH3 Auto Detection",string(32))
25 $vlEth3V1("ETH 3 Video VLAN 1 (VID or COS,VID)",string(32))
26 $autoDetectConfigEth4("ETH4 Auto Detection",string(32))
27
28
29 $vlanPots("Voice VLAN",string(32))
30 $voipIpOut("VOIP Host IP Option: 2-static, 3-DHCP",string(32))

```

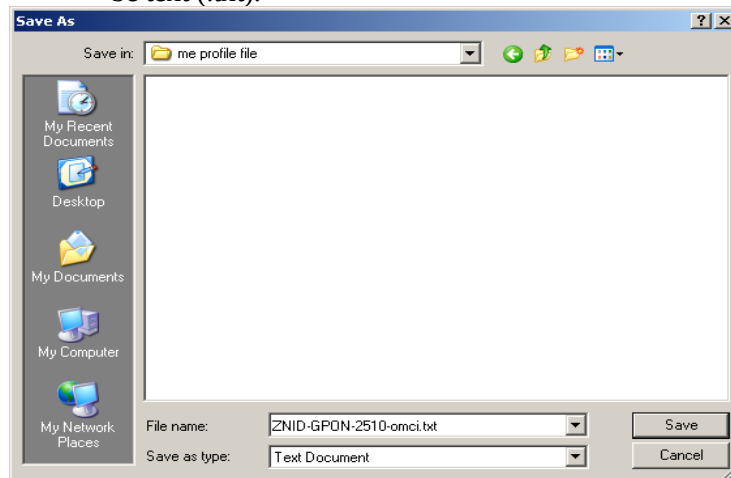
- 7 Two options are displayed on the top of the ME profile file page, Edit Config and Download Config.
- Clicking on the **Edit Config** button causes the web-interface to return to the service page. This page lists the current selection. You can change the configuration, and create a new ME profile file.

- Clicking on the **Download Config** button causes the web-interface to display a File Download window.



Click **Save** to open the Save As window.

In the Save As window, changes the path and chooses the appropriate file name to save the newly created ME profile file. The file type must be text (.txt).



Downloading ME profile file to MXK

The service provider needs to download ME profile file from a TFTP/ SFTP server to MXK.

- 1 Create a directory at the root level, then download the ME profile file.

In this example the directory is named as *me*.

There are no restrictions on the directory name.

```
zSH> mkdir /me
```

- 2 Download the ME profile file to the MXK with the **file download** command. This example downloads the ME profile file *ZNID-GPON-2510-omci.txt* from a TFTP server *172.16.80.201* to the MXK */me* directory, and save the ME profile file with the same name.

```
zSH> file download 172.16.80.201 /
ZNID-GPON-2510-omci.txt /me/ZNID-GPON-2510-omci.txt
File download successful
```

Creating ME profile

Create an ME profile from the ME profile file. One ME profile is created for each ONU model.

The software supports a text import capability to read the ME profile file and learn the ME structure of the new ONU. The ME profile contains OMCI ME commands.

- 1 Create an ME profile:

```
zSH> gpononu profiles create me me1 /me/
ZNID-GPON-2510-omci.txt
Profile created
```

- 2 Verify the created ME profile name:

```
zSH> gpononu profiles show me
me1
```

Creating Generic profiles

The Generic profile defines the values of variables that define service plans. It also contains values of system variables. The system values, service plan values are entered by the service provider as part of system commissioning.

If the service provider intend to offer 3 different service plans that are supported on 5 different ONU hardware models, service provider should create 5 ME profiles and 15 Generic profiles in the system.

To create a Generic profile:

- 1 Create a Generic profile:

```
zSH> gpononu profiles create gen gen1 me1
Profile created
```

- 2 Verify the created Generic profile name.

```
zSH> gpononu profiles show gen
gen1
```


A Generic profile contains the variables that are typically parameters that associated with the specific service plan. For example, voice vlan. The variables can also be parameters that are generic to the system. For example, Softswitch IP address (i.e. the parameter SIP Proxy IP in the following generic profile example).

- 3 Assign values to the desired variables in the Generic profile with the **gpononu profiles update gen** command and enter the corresponding variable indexes.

```

zSH> gpononu profiles update gen gen1
Generic Profile: gen1
 1 "ETH1 Auto Detection [0]"
 2 "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"
 3 "ETH2 Auto Detection [0]"
 4 "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"
 5 "ETH3 Auto Detection [0]"
 6 "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
 7 "ETH4 Auto Detection [0]"
 8 "Voice VLAN [7,200]"
 9 "VOIP Host IP Option: 2-static, 3-DHCP [2]"
10 "VOIP Host IP [0.0.0.0]"
11 "VOIP Netmask [0.0.0.0]"
12 "VOIP Gateway [0.0.0.0]"
13 "VOIP Server IP [0.0.0.0]"
14 "VOIP Primary DNS [0.0.0.0]"
15 "VOIP Secondary DNS [0.0.0.0]"
16 "Country Code [ 1]"
17 "Rx Gain [0]"
18 "Tx Gain [0]"
19 "Out-of-band DTMF [0]"
20 "Echo Cancel: 1-enable, 0-disable [1]"
21 "POTS1 Dial Number [1111]"
22 "POTS1 User Name [1111]"
23 "POTS1 Password [1111]"
24 "POTS2 Dial Number [2222]"
25 "POTS2 User Name [2222]"
26 "POTS2 Password [2222]"
27 "Fax Mode [0]"
28 "CID Features [63]"
29 "Call Waiting Features [3]"
30 "Call Progress or Transfer Features [255]"
31 "Call Present Features [15]"
32 "ETH 1 Admin Status: 0-Up, 1-Down [0]"
33 "ETH 2 Admin Status: 0-Up, 1-Down [0]"
34 "ETH 3 Admin Status: 0-Up, 1-Down [0]"
35 "ETH 4 Admin Status: 0-Up, 1-Down [0]"
36 "POTS 1 Admin Status: 0-Up, 1-Down [0]"
37 "POTS 2 Admin Status: 0-Up, 1-Down [0]"
Enter OMCI edit command or [s]ave, [q]uit, [h]elp: h

```

Available Commands:

- E - display edit data (short)
- H - display help

```

L    - display edit data (long)
Q    - quit without save
S    - save and exit
1..n - edit variable #n

```

```

Enter OMCI edit command or [s]ave, [q]uit, [h]elp: 1
Enter value: 1
Enter OMCI edit command: 3
Enter value: 1
...
Enter OMCI edit command: s
GENERIC profile has been saved

```

- 4 View additional edit information for the variables in the Generic profile with the **gpononu profiles update gen** command and enter OMCI edit command **L** (not case sensitive).

```

zSH> gpononu profiles update gen gen1
Generic Profile: gen1
 1 "ETH1 Auto Detection [1]"
 2 "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"
 3 "ETH2 Auto Detection [0]"
 4 "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"
 5 "ETH3 Auto Detection [1]"
 6 "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
 7 "ETH4 Auto Detection [0]"
 8 "Voice VLAN [7,200]"
 9 "VOIP Host IP Option: 2-static, 3-DHCP [2]"
10 "VOIP Host IP [0.0.0.0]"
11 "VOIP Netmask [0.0.0.0]"
12 "VOIP Gateway [0.0.0.0]"
13 "VOIP Server IP [0.0.0.0]"
14 "VOIP Primary DNS [0.0.0.0]"
15 "VOIP Secondary DNS [0.0.0.0]"
16 "Country Code [ 1]"
17 "Rx Gain [0]"
18 "Tx Gain [0]"
19 "Out-of-band DTMF [0]"
20 "Echo Cancel: 1-enable, 0-disable [1]"
21 "POTS1 Dial Number [1111]"
22 "POTS1 User Name [1111]"
23 "POTS1 Password [1111]"
24 "POTS2 Dial Number [2222]"
25 "POTS2 User Name [2222]"
26 "POTS2 Password [2222]"
27 "Fax Mode [0]"
28 "CID Features [63]"
29 "Call Waiting Features [3]"
30 "Call Progress or Transfer Features [255]"
31 "Call Present Features [15]"
32 "ETH 1 Admin Status: 0-Up, 1-Down [0]"
33 "ETH 2 Admin Status: 0-Up, 1-Down [0]"
34 "ETH 3 Admin Status: 0-Up, 1-Down [0]"
35 "ETH 4 Admin Status: 0-Up, 1-Down [0]"

```

```

36 "POTS 1 Admin Status: 0-Up, 1-Down [0]"
37 "POTS 2 Admin Status: 0-Up, 1-Down [0]"
Enter OMCI edit command or [s]ave, [q]uit, [h]elp: 1

ID      Generic Profile: gen1
=====
=====
1  Name      : $autoDetectConfigEth1
   Comment   : ETH1 Auto Detection
   Type      : string(32)
   Gen Value :
   Default Value: 0
2  Name      : $vLEth1V1
   Comment   : ETH 1 Data VLAN 1 (VID or COS,VID)
   Type      : string(32)
   Gen Value :
   Default Value: 0,100
3  Name      : $autoDetectConfigEth2
   Comment   : ETH2 Auto Detection
   Type      : string(32)
   Gen Value :
   Default Value: 0
<SPACE> for next page, <CR> for next line, A for all, Q
to quitq

```

Creating Specific profiles

For each end-user, the service provider creates a Specific profile. The Specific profile is similar to the Generic profile, it is also a collection of variables that are used by an ME profile.

Every Specific profile is associated with only one ME profile and one Generic profile to define the supported ONU hardware model and service plan. The Specific profile provides values for variables that are specific to the ONU.

- 1 Create an end-user based Specific profile with the **gpononu profiles create spec** *slot/olt/onu meProfileName genericProfileName* command:

```

zSH> gpononu profiles create spec 13/1/1 me1 gen1
Profile created

```

- 2 Verify the name of the Specific profile created on ONU 13/1/1.

```

zSH> gpononu profiles show spec 13/1/1
13/1/1

```

- 3 View the references to the ME profile and Generic profile on ONU 13/1/1:

```

zSH> gpononu show 13/1/1
Slot 13 olt 1

```

ONU	Name	Enabled	Serial Number	OMCI files and profiles
===	=====	=====	=====	=====

1 1-13-1-1

No

ME me1

GEN gen1

- 4 Assign values to the variables that are unique to the end-user, such as user's IP address and telephone number etc.

zSH> **gpononu profiles update spec 13/1/1**

Specific Profile: 13/1/1

```

1 "ETH1 Auto Detection [1]"
2 "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"
3 "ETH2 Auto Detection [1]"
4 "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"
5 "ETH3 Auto Detection [0]"
6 "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
7 "ETH4 Auto Detection [0]"
8 "Voice VLAN [7,200]"
9 "VOIP Host IP Option: 2-static, 3-DHCP [2]"
10 "VOIP Host IP [0.0.0.0]"
11 "VOIP Netmask [0.0.0.0]"
12 "VOIP Gateway [0.0.0.0]"
13 "VOIP Server IP [0.0.0.0]"
14 "VOIP Primary DNS [0.0.0.0]"
15 "VOIP Secondary DNS [0.0.0.0]"
16 "Country Code [ 1]"
17 "Rx Gain [0]"
18 "Tx Gain [0]"
19 "Out-of-band DTMF [0]"
20 "Echo Cancel: 1-enable, 0-disable [1]"
21 "POTS1 Dial Number [1111]"
22 "POTS1 User Name [1111]"
23 "POTS1 Password [1111]"
24 "POTS2 Dial Number [2222]"
25 "POTS2 User Name [2222]"
26 "POTS2 Password [2222]"
27 "Fax Mode [0]"
28 "CID Features [63]"
29 "Call Waiting Features [3]"
30 "Call Progress or Transfer Features [255]"
31 "Call Present Features [15]"
32 "ETH 1 Admin Status: 0-Up, 1-Down [0]"
33 "ETH 2 Admin Status: 0-Up, 1-Down [0]"
34 "ETH 3 Admin Status: 0-Up, 1-Down [0]"
35 "ETH 4 Admin Status: 0-Up, 1-Down [0]"
36 "POTS 1 Admin Status: 0-Up, 1-Down [0]"
37 "POTS 2 Admin Status: 0-Up, 1-Down [0]"

```

Enter OMCI edit command: **21**

Enter value: **5105551000**

Enter OMCI edit command: **22**

Enter value: **5105551000**

...

Enter OMCI edit command: **s**

SPECIFIC profile has been saved

- 5 View the configured specific profile:

```

zSH> gpononu profiles update spec 13/1/1
Specific Profile: 13/1/1
 1 "ETH1 Auto Detection [1]"
 2 "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"
 3 "ETH2 Auto Detection [1]"
 4 "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"
 5 "ETH3 Auto Detection [0]"
 6 "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
 7 "ETH4 Auto Detection [0]"
 8 "Voice VLAN [7,200]"
 9 "VOIP Host IP Option: 2-static, 3-DHCP [2]"
10 "VOIP Host IP [0.0.0.0]"
11 "VOIP Netmask [0.0.0.0]"
12 "VOIP Gateway [0.0.0.0]"
13 "VOIP Server IP [0.0.0.0]"
14 "VOIP Primary DNS [0.0.0.0]"
15 "VOIP Secondary DNS [0.0.0.0]"
16 "Country Code [ 1]"
17 "Rx Gain [0]"
18 "Tx Gain [0]"
19 "Out-of-band DTMF [0]"
20 "Echo Cancel: 1-enable, 0-disable [1]"
21 "POTS1 Dial Number [5105551000]"
22 "POTS1 User Name [5105551000]"
23 "POTS1 Password [11111]"
24 "POTS2 Dial Number [2222]"
25 "POTS2 User Name [22222]"
26 "POTS2 Password [22222]"
27 "Fax Mode [0]"
28 "CID Features [63]"
29 "Call Waiting Features [3]"
30 "Call Progress or Transfer Features [255]"
31 "Call Present Features [15]"
32 "ETH 1 Admin Status: 0-Up, 1-Down [0]"
33 "ETH 2 Admin Status: 0-Up, 1-Down [0]"
34 "ETH 3 Admin Status: 0-Up, 1-Down [0]"
35 "ETH 4 Admin Status: 0-Up, 1-Down [0]"
36 "POTS 1 Admin Status: 0-Up, 1-Down [0]"
37 "POTS 2 Admin Status: 0-Up, 1-Down [0]"
Enter OMCI edit command: q
SPECIFIC profile modifications have been discarded

```

Viewing current variable settings on an ONU



Note: Make sure every configuration variable on the ONU has value assigned. Otherwise configuration fails for this ONU unless you updates the Generic profile or Specific profile to provide a value.

View the current settings of configuration variables on an ONU by using the **gpononu profiles show vars slot/olt/onu | interfaceName** command. This command lists all variables defined in the ME profile. For each variable, it displays description, value, and source. The description comes from the

commands in the ME profile. Value is what currently being used. Source indicates where this value came from (Generic, ME, or Specific profile).

- If the values of all the variables is not available in the Specific profile it checks the Generic profile. The Specific profile can be used to override any values set in the Generic profile. The Generic profile can override any values set in the ME profile.
- If the value is only defined in ME profile, “Default” will be displayed in the Source column.
- If there is no default or other value given, “NONE” will be displayed in the Source column, “NO VALUE ASSIGNED” will be displayed in the Value column. In that case, configuration fails for this ONU unless updates the Generic profile or Specific profile to provide a value.

View current settings of configuration variables on ONU 13/1/1, enter:

zSH> gpononu profiles show vars 13/1/1

Variable	Description	Value	Source
1	ETH1 Auto Detection	1 (10BaseT Full)	Generic
2	ETH 1 Data VLAN 1 (VID or COS,VID)	0,100	Default
3	ETH2 Auto Detection	1 (10BaseT Full)	Generic
4	ETH 2 Data VLAN 1 (VID or COS,VID)	0,100	Default
5	ETH3 Auto Detection	0 (Auto-sensing)	Default
6	ETH 3 Video VLAN 1 (VID or COS,VID)	4,999	Default
7	ETH4 Auto Detection	0 (Auto-sensing)	Default
8	Voice VLAN	7,200	Default
9	VOIP Host IP Option: 2-static, 3-DHCP	2	Default
10	VOIP Host IP	0.0.0.0	Default
11	VOIP Netmask	0.0.0.0	Default
12	VOIP Gateway	0.0.0.0	Default
13	VOIP Server IP	0.0.0.0	Default
14	VOIP Primary DNS	0.0.0.0	Default
15	VOIP Secondary DNS	0.0.0.0	Default
16	Country Code	1	Default
17	Rx Gain	0	Default
18	Tx Gain	0	Default
19	Out-of-band DTMF	0	Default
20	Echo Cancel: 1-enable, 0-disable	1	Default
21	POTS1 Dial Number	5105551000	Specific
22	POTS1 User Name	5105551000	Specific
23	POTS1 Password	11111	Default
24	POTS2 Dial Number	2222	Default
25	POTS2 User Name	22222	Default
26	POTS2 Password	22222	Default
27	Fax Mode	0	Default
28	CID Features	63	Default
29	Call Waiting Features	3	Default
30	Call Progress or Transfer Features	255	Default
31	Call Present Features	15	Default
32	ETH 1 Admin Status: 0-Up, 1-Down	0 (up)	Default
33	ETH 2 Admin Status: 0-Up, 1-Down	0 (up)	Default

34	ETH 3	Admin Status: 0-Up, 1-Down	0 (up)	Default
35	ETH 4	Admin Status: 0-Up, 1-Down	0 (up)	Default
36	POTS 1	Admin Status: 0-Up, 1-Down	0 (up)	Default
37	POTS 2	Admin Status: 0-Up, 1-Down	0 (up)	Default

Activating an ONU with OMCI profiles



Note: Make sure to provision the logical connections for data, video, and voice services in the MXK and ONUs before activating the ONUs in order to avoid having to re-sync or reboot the ONUs eventually.

Refer to [Create a GEM port with a GPON traffic profile, page 451](#) to get detail configuration.

Activate an ONU with serial number and OMCI profiles association with this command:

```
gpononu set <slot/olt/onu| interfaceName> <sernoID| vendorid vendorID|
serno fsan a hex number| a decimal number> meprof meProfileName
genprof genericProfileName
```

During activation process, when the OLT communicates with the ONU, the OLT retrieves current variable settings on this ONU.

To activate ONU 1/13/1/1, perform the following tasks:

- 1 Display the line status on shelf 1, slot 13 with the **showline shelfID/slotID** command. OOS (Out of Service) means this ONU is not activated.

```
zSH> showline 1/13
```

```
Search in progress .....
```

```
-----
shelf = 1, slot = 13, port 1, line type = ONU
```

```
subport
```

```
  1-12    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
  13-24    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
  25-36    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
```

```
-----
shelf = 1, slot = 13, port 2, line type = ONU
```

```
subport
```

```
  1-12    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
  13-24    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
  25-36    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
```

```
-----
shelf = 1, slot = 13, port 3, line type = ONU
```

```
subport
```

```
  1-12    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
  13-24    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
```

```
25-36    OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS  OOS
```

```
-----
shelf = 1, slot = 13, port 4, line type = ONU
```

```
subport
```

```

1-12    00S 00S 00S 00S 00S 00S 00S 00S 00S 00S 00S 00S 00S
13-24   00S 00S 00S 00S 00S 00S 00S 00S 00S 00S 00S 00S 00S
25-36   00S 00S 00S 00S 00S 00S 00S 00S 00S
-----
shelf = 1, slot = 13, line type = OLT
line
1-12    ACT ACT ACT ACT
```

2 View all free ONUs and available serial numbers under Slot 13 OLT 1:

```

zSH> gpononu show 13/1
Free ONUs for slot 13 olt 1:
  1  17  18  19  20  21  22  23  24  25  26  27 28 29 30 31 32
Discovered serial numbers for slot 13 olt 1:
sernoID  Vendor  Serial Number      sernoID  Vendor  Serial Number
  1      ZNTS    1306
```

3 Assign ONU 13/1/1 with sernoID 1 (the sernoID 1 is associated with serial number 1306), ME profile *me1*, and Generic profile *gen1*.

```

zSH> gpononu set 13/1/1 1 meprofile me1 genprofile gen1
Onu 1 successfully enabled with serial number ZNTS 1306
```



Note: If configuring an ONU with flat OMCI (i.e. OMCI file) not with Smart OMCI, assign the available ONU # with *sernoID* and modify the bandwidth on the GEM ports associated with the ONU, then associate an OMCI file with the ONU, enter **gpononu set slot/olt/onu sernoID bw gemport/bwvalue omci filename**.

The default ONU bandwidth is 1 Mbps. You may want to associate up to three GEM ports with ONU and modify the bandwidth to provide a subscriber with various services requiring different bandwidths such as voice, video, and data.

```

zSH> gpononu set 4/2/1 3 bw 501/10 bw 701/2
bw 901/1 omci xavi110p_data_video_data
Bandwidth for 501 is 10 Mbps.
Bandwidth for 701 is 2 Mbps.
Bandwidth for 901 is 1 Mbps.
Onu 1 successfully enabled with serial
number XAVI 2415919203
```

4 Verify ONU 13/1/1 is activated with serial number 1306 and OMCI profiles *me1* and *gen1*.

```

zSH> gpononu show 13/1/1

Slot 13 olt 1

ONU      Name      Enabled      Number      OMCI files and profiles
=== =====
```



```

1          1-13-1-1   Yes    ZNTS 1306   ME    me1
                                GEN    gen1

```

If you want to view all the OLTs and ONUs on the MXK, use the **gpononu showall** command.

You can use the **gpononu showall slot/olt** command to display the OLT port to which the ONU is connected, whether the ONU is enabled, the ONU Model number, the serial number of the ONU, and the associated OMCI profiles.

```

zSH> gpononu showall 13/1
Processing list of 64

```

This command may take several minutes to complete.

Do you want to continue? (yes or no) [~~no~~yes]

ONU	Name	Enabled	Model #	Serial Number	OMCI files and profiles
1	1-13-1-1	Yes	2510	ZNTS 1306 ME	me1 GEN gen1
2	1-13-1-2	No			(none)

<SPACE> for next page, <CR> for next line, A for all, Q to quit

View the status of a specific ONU with the **gpononu showall slot/olt/onu** command.

```

zSH> gpononu showall 13/1/1

```

ONU	Name	Enabled	Model #	Serial Number	OMCI files and profiles
1	1-13-1-1	Yes	2510	ZNTS 1306 ME	me1 GEN gen1

View all enabled ONUs on an OLT with the **gpononu showall slot/olt enabled** command:

```

zSH> gpononu showall 13/1 enabled
Processing list of 64

```

This command may take several minutes to complete.

Do you want to continue? (yes or no) [~~no~~yes]

ONU	Name	Enabled	Model #	Serial Number	OMCI files and profiles
1	1-13-1-1	Yes	2510	ZNTS 1306 ME	me1 GEN gen1

<SPACE> for next page, <CR> for next line, A for all, Q to quit

- 5 Disable an ONU and clear the serial number for this ONU (if necessary), use the **gpononu clear** command.

```
zSH> gpononu clear 13/1/1
```

Onu1 (previously with serial number ZNT 1306) has been cleared

To also delete the ME and Generic profile references, and delete the Specific profile on this ONU, use the **gpononu clear** command with option **omci**.

```
zSH> gpononu clear 13/1/1 omci
```

Onu1 (previously with serial number ZNT 1306) has been cleared

Delete OMCI profile

This section describes how to delete ME profile, Generic profile and Specific profile.

- The Specific profile can be deleted when the associated ONU is either activated or not activated.

Note that without the Specific profile, the OMCI provisioning on the associated ONU will be disabled.

- The ME profile and Generic profile can be deleted when they are not being used. Otherwise, the error message will be displayed stating this profile is being used.
 - The ME profile used could have a Generic profile and/or Specific profile associated with it. In that case, remove the related Generic and/or Specific profile first, and then delete the ME profile.
 - The Generic profile used could have a Specific profile associated with it. In that case, remove the related Specific profile, then delete the Generic profile.
 - An ONU is associated with this ME profile or Generic profile. In that case, remove the ME profile or Generic profile references from the ONU, then delete the ME profile or Generic profile.

Two different commands are provided to remove the ME/Generic profile references from an non-activated ONU or an activated ONU:

gpononu set noomci command

gpononu clear omci command

Deleting the OMCI profile when the ONU is not activated

This section describes how to delete the ME, Generic, Specific profile when the associated ONU is not activated.

This example assumes the ME profile *me1* has one Generic profile, *gen1*, and one Specific profile, *13/1/1*, associated with it:

- 1 Verify ONU 13/1/1 is not active, and the ME profile *me1* and Generic profile *gen1* are associated with ONU 13/1/1.

```
zSH> gpononu show 13/1/1
Slot 13 olt 1
```

ONU	Name	Enabled	Serial Number	OMCI files and profiles
1	1-13-1-1	No		ME me1 GEN gen1

- 2 Delete the ME profile and Generic profile references to the Specific profile, and delete the Specific profile created on ONU 13/1/1 with the **gpononu set noomci** command.

This command does not change the state of the existing ONU.

Note that this command should not be used under the following two conditions:

- if the OMCI profiles were not previously set.
- if the specified ONU is currently active.

```
zSH> gpononu set 13/1/1 noomci
```

- 3 Verify the ME profile name and Generic profile name are removed from ONU 13/1/1.

```
zSH> gpononu show 13/1/1
Slot 13 olt 1
```

ONU	Name	Enabled	Serial Number	OMCI files and profiles
1	1-13-1-1	No		(none)

- 4 Verify the Specific profile on ONU 13/1/1 is also removed.

```
zSH> gpononu profiles show spec 13/1/1
```

The outputs does not show 13/1/1 indicating a Specific profile created on 13/1/1 does not exist.

- 5 Delete the Generic profile, then delete the ME profile.

```
zSH> gpononu profiles delete gen gen1
```

Profile has been deleted!

```
zSH> gpononu profiles delete me me1
```

Profile has been deleted!

Deleting the OMCI profile when the ONU is activated

This section describes how to delete a Specific profile, Generic profile and ME profile on an activated ONU.

The following examples assume ME profile *me1* has one Generic profile, *gen1*, and one Specific profile, *13/1/1*, associated with it:

- 1 Delete a Specific profile that is used by an activated ONU. The OMCI configuration state on this ONU is changed after deleting Specific profile.
 - a Verify the Specific profile associated ONU is activated. And the *OmciConfigState* is Done.

```
zSH> gpononu show 13/1/1
```

```
Slot 13 olt 1
```

ONU	Name	Enabled	Number	OMCI files and profiles
1	1-13-1-1	Yes	ZNTS 1306	ME me1 GEN gen1

```
zSH> gpononu status 13/1/1
```

```
Slot 13 olt 1
```

ID	Onu	OperStatus	OmciConfigState	GponOnuStatus
1	1-13-1-1	Up	Done	Active

- b Delete the Specific profile created on this ONU.

```
zSH> gpononu profiles delete spec 13/1/2
```

```
Profile has been deleted!
```

- c Without the Specific profile, the OMCI provisioning on the associated ONU will be disabled. Verify the *OmciConfigState* is not Done.

```
zSH> gpononu status 13/1/1
```

```
Slot 13 olt 1
```

ID	Onu	OperStatus	OmciConfigState	GponOnuStatus
1	1-13-1-1	Up	Config	Active

- 2 Delete an ME profile and a Generic profile that is used by an activated ONU.

- a Deleting an ME profile and Generic profile that is used by an ONU causes an error message to appear.

```
zSH> gpononu profiles delete me me1
```

```
ERROR! Cannot delete, profile is being used
```

```
zSH> gpononu profiles delete gen gen1
```

ERROR! Cannot delete, profile is being used

- b** Clear the serial number of the ONU, delete the ME profile and Generic profile references, and the Specific profile (if any) and disable the ONU with the **gpononu clear omci** command:

```
zSH> gpononu clear 13/1/1 omci
```

Verify that the ME profile name and Generic profile name are removed from ONU 13/1/1, and that ONU is disabled.

```
zSH> gpononu show 13/1/1
Slot 13 olt 1
```

ONU	Name	Enabled	Serial Number	OMCI files and profiles
1	1-13-1-1	No		(none)

Verify the Specific profile on ONU 13/1/1 is also removed, enter.

```
zSH> gpononu profiles show spec 13/1/1
```

The outputs didn't show 13/1/1 indicating that the Specific profile does not exist on 13/1/1.

- c** Delete Generic profile, then delete ME profile.

```
zSH> gpononu profiles delete gen gen1
```

Profile has been deleted!

```
zSH> gpononu profiles delete me me1
```

Profile has been deleted!

Import and export OMCI profile

Importing OMCI profile

The OMCI profile import feature allows the contents of an existing OMCI profile to be overwritten with a new OMCI profile file.

Any changes in the OMCI profile file, such as adding or deleting OMCI commands in ME profile file, will cause the variables in the OMCI profile to be added, deleted, or remain the same. After importing the OMCI profile file to the existing OMCI profile, system will reconcile the associated Generic profile and Specific profile. User can update the variables in the Generic and Specific profile as needed.

To import a new OMCI (ME, Generic, or Specific) profile file to an existing OMCI profile, use the following commands:

- **gpononu profiles import me** *meProfileName fileName* command
- **gpononu profiles import gen** *genProfileName fileName* command

- **gpononu profiles import spec** *slot/olt/onu fileName* command

The following example shows how to import an ME profile file and related configuration:

- 1 View the existing ME profiles.

```
zSH> gpononu profiles show me
me1
me2
```

- 2 Find the ONUs that use the selected ME profile.

```
zSH> gpononu profiles find me me1
This command may take several minutes to complete.
Do you want to continue? <yes or no> [no] yes

13/1/1
```

The above example shows only ONU 13/1/1 uses *me1*.

- 3 Import the new ME profile file to the ME profile *me1*.

Importing the new ME profile file overwrites the current contents in the *me1*, and a warning message appears.

```
zSH> gpononu profiles import me me1 /me/2510-mev2.txt
Profile imported.
```

Variables may have been added, deleted, or changed in the ME Profile "me1". The Generic and Specific profiles associated with the ME profile "me1" have been reconciled to include these variable modifications (if any). Please edit the relevant Generic and Specific profiles accordingly.

- 4 Find the relevant Generic profile, and then specify the desired values to the variables in the Generic profile.

```
zSH> gpononu show 13/1/1
```

```
Slot 13 olt 1
```

ONU	Name	Enabled	Number	OMCI files and profiles
1	1-13-1-1	Yes	ZNTS 1306	ME me1 GEN gen1

The above example shows ONU 13/1/1 uses *gen1*. Then update the generic profile *gen1* as desired.

```
zSH> gpononu profiles update gen gen1
```

Generic Profile: gen1

```

1  "newvariable"                                the new variable
2  "ETH1 Auto Detection [1]"
3  "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"
4  "ETH2 Auto Detection [0]"
5  "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"
6  "ETH3 Auto Detection [1]"
7  "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
8  "ETH4 Auto Detection [0]"
9  "Voice VLAN [7,200]"
10 "VOIP Host IP Option: 2-static, 3-DHCP [2]"
11 "VOIP Host IP [0.0.0.0]"
12 "VOIP Netmask [0.0.0.0]"
13 "VOIP Gateway [0.0.0.0]"
14 "VOIP Server IP [0.0.0.0]"
15 "VOIP Primary DNS [0.0.0.0]"
16 "VOIP Secondary DNS [0.0.0.0]"
17 "Country Code [ 1]"
18 "Rx Gain [0]"
19 "Tx Gain [0]"
20 "Out-of-band DTMF [0]"
21 "Echo Cancel: 1-enable, 0-disable [1]"
22 "POTS1 Dial Number [1111]"
23 "POTS1 User Name [1111]"
24 "POTS1 Password [1111]"
25 "POTS2 Dial Number [2222]"
26 "POTS2 User Name [2222]"
27 "POTS2 Password [2222]"
28 "Fax Mode [0]"
29 "CID Features [63]"
30 "Call Waiting Features [3]"
31 "Call Progress or Transfer Features [255]"
32 "Call Present Features [15]"
33 "ETH 1 Admin Status: 0-Up, 1-Down [0]"
34 "ETH 2 Admin Status: 0-Up, 1-Down [0]"
35 "ETH 3 Admin Status: 0-Up, 1-Down [0]"
36 "ETH 4 Admin Status: 0-Up, 1-Down [0]"
37 "POTS 1 Admin Status: 0-Up, 1-Down [0]"
38 "POTS 2 Admin Status: 0-Up, 1-Down [0]"
Enter OMCI edit command: 1
Enter value: ###
...
Enter OMCI edit command: s
GENERIC profile has been saved

```

- 5 Specify the desired values to the variables in the relevant Specific profile.

zSH> **gpononu profiles update spec 13/1/1**

Specific Profile: 13/1/1

```

1"newvariable"                                the new variable
2 "ETH1 Auto Detection [1]"
3 "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"
4 "ETH2 Auto Detection [0]"
5 "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"

```

```

6 "ETH3 Auto Detection [1]"
7 "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
8 "ETH4 Auto Detection [0]"
9 "Voice VLAN [7,200]"
10 "VOIP Host IP Option: 2-static, 3-DHCP [2]"
11 "VOIP Host IP [0.0.0.0]"
12 "VOIP Netmask [0.0.0.0]"
13 "VOIP Gateway [0.0.0.0]"
14 "VOIP Server IP [0.0.0.0]"
15 "VOIP Primary DNS [0.0.0.0]"
16 "VOIP Secondary DNS [0.0.0.0]"
17 "Country Code [ 1]"
18 "Rx Gain [0]"
19 "Tx Gain [0]"
20 "Out-of-band DTMF [0]"
21 "Echo Cancel: 1-enable, 0-disable [1]"
22 "POTS1 Dial Number [1111]"
23 "POTS1 User Name [1111]"
24 "POTS1 Password [1111]"
25 "POTS2 Dial Number [2222]"
26 "POTS2 User Name [2222]"
27 "POTS2 Password [2222]"
28 "Fax Mode [0]"
29 "CID Features [63]"
30 "Call Waiting Features [3]"
31 "Call Progress or Transfer Features [255]"
32 "Call Present Features [15]"
33 "ETH 1 Admin Status: 0-Up, 1-Down [0]"
34 "ETH 2 Admin Status: 0-Up, 1-Down [0]"
35 "ETH 3 Admin Status: 0-Up, 1-Down [0]"
36 "ETH 4 Admin Status: 0-Up, 1-Down [0]"
37 "POTS 1 Admin Status: 0-Up, 1-Down [0]"
38 "POTS 2 Admin Status: 0-Up, 1-Down [0]"
Enter OMCI edit command: 1
Enter value: ###
...
Enter OMCI edit command: s
SPECIFIC profile has been saved

```

- 6 Send the latest OMCI commands from OLT to ONU.

```
zSH> gpononu resync 13/1/1
```

Exporting OMCI profile file

The OMCI profile export feature allows user to copy the contents of an OMCI profile to a new OMCI profile file, and store this file into the uplink compact flash. Later this OMCI profile file could be shared with other devices.

To export the contents of an OMCI (ME, Generic, or Specific) profile to a new OMCI profile file, use the following commands:

- **gpononu profiles export me** *meProfileName fileName* command
- **gpononu profiles export gen** *genProfileName fileName* command

- **gpononu profiles export spec slot/olt/onu fileName** command

The following example copies an ME profile file from an MXK to another MXK:

- 1 On MXK A, copy the contents of ME profile *me1* to a new ME profile file, name it as *2510-mev1.txt*, and save it to *root/pub* directory.

```
zSH> cd pub
```

```
zSH> gpononu profiles export me me1 2510-mev1.txt
Profile exported.
```

By default, the MXK runs as a TFTP server enabling files stored in the *root/pub* folder to be downloaded to other devices with connectivity to the MXK.

- 2 On MXK B, download the ME profile file *2510-mev1.txt* from the MXK A (IP address 172.42.15.19) to the local directory *me*, and name it as *2510-me.txt*.

```
zSH> file download 172.42.15.19 /pub/2510-mev1.txt /
me/2510-me.txt
File download successful
```

- 3 Import the ME profile file *2510-me.txt* to an existing ME profile *2510me*.

```
zSH> gpononu profiles import me 2510me /me/2510-me.txt
Profile imported.
```

Variables may have been added, deleted, or changed in the ME Profile "me1". The Generic and Specific profiles associated with the ME profile "me1" have been reconciled to include these variable modifications (if any). Please edit the relevant Generic and Specific profiles accordingly.

- 4 Update the Generic profile or Specific profile as desired.

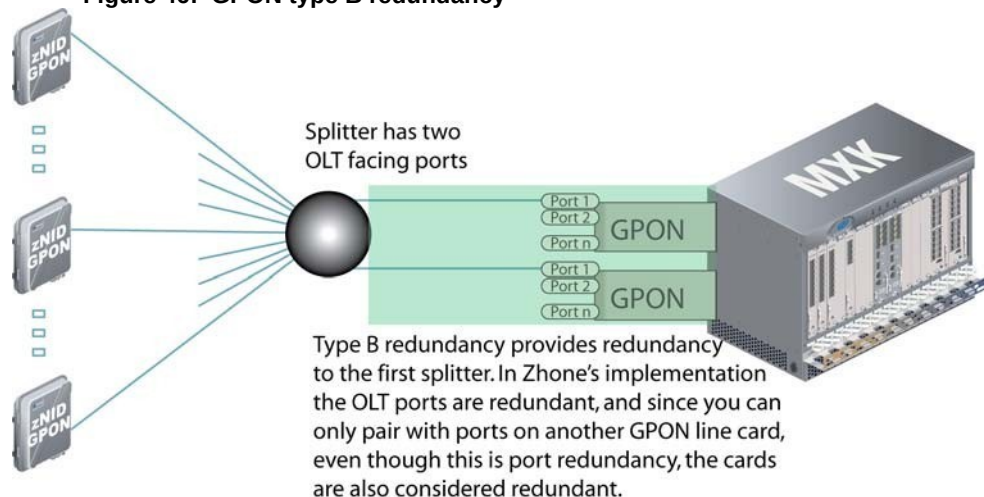
See steps 4 to 6 in [Importing OMCI profile on page 413](#).

GPON type B redundancy

The MXK supports GPON type B redundancy as specified in the ITU-T G.984.1 standards specification.

Type B GPON redundancy doubles both the OLT ports on the GPON line card and the optical fiber between the OLT ports and the optical splitter which is closest to the OLTs. You must use a splitter with two input/output ports on the OLT side. Outages on fiber from the OLT to the first splitter can be recovered from; With Type B redundancy there is no redundancy from the splitter to the ONT.

Figure 46: GPON type B redundancy



With Zhone's GPON type B redundancy, a GPON redundancy group can have two GPON OLT ports and the two GPON OLT ports must be on different GPON line cards. The ports can be on a 4 port or 8 port GPON line card. So even though the cards themselves are not redundant, their ports may be. Two 4 port GPON line cards can provide redundancy for a single 8 port GPON line card. Since it is port level redundancy and not card level redundancy, the port numbers on one card do not need to match the port number on the second card.

A single GPON port cannot be configured in two groups at the same time.

When the GPON ports in a GPON redundancy group are added, the active and standby port are based on whether they are added as a primary or secondary interface in the **line-red add** command. If you reboot the MXK system (or reboot both cards which have the redundant ports), the OLT port which comes up first and is able to pass traffic will be the active port.

In a redundancy group one OLT port is always assigned as active and the other standby. If an active OLT port fails, the standby takes over and becomes active. Note that OLT redundancy is non-revertive; that is, a previously active OLT port which has failed does not become active when the reason for the failover is resolved. The current active port will stay active until that port/line fails, then the standby (if the initial issue was resolved) will once again become the active port.

When a standby port is added to the redundancy group and comes up, the card with the active port copies over the configuration database and routing tables to the standby OLT port on the second card. As configuration changes are made to the active port, the standby port is automatically updated.



Note: Create the line redundancy before building interfaces. If you add a port with existing interfaces as primary port of the redundant pair, you will need to perform a **slot reboot** on the GPON card with the secondary port after adding the redundancy.

Configuring GPON line redundancy

1 Show that there is no redundancy

You can verify whether redundancy has been added by using the **gponolt show redund** command:

```
zSH> gponolt show redund
```

OLT	Interface	Status	Redundancy State	Redundancy Peer OLT	Redundancy Peer Interface
3/1	1-3-1-0	OOS			
3/2	1-3-2-0	OOS			
3/3	1-3-3-0	OOS			
3/4	1-3-4-0	UP			

OLT	Interface	Status	Redundancy State	Redundancy Peer OLT	Redundancy Peer Interface
4/1	1-4-1-0	OOS			
4/2	1-4-2-0	OOS			
4/3	1-4-3-0	OOS			
4/4	1-4-4-0	UP			

This display shows no redundancy because there is no information showing in the **Redundancy State** or the **OLT** and **Interface** columns.

You can also show redundancy for a specific port using the **line-red show interface/physical interface** type command:

```
zSH> line-red show 1-3-4-0
```

```
redundancy status for 1-3-4-0/gponolt:
```

```
NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0
```

```
The 1-3-4-0/gponolt is not part of any redundancy group
```

2 Add line redundancy

```
zSH> line-red add pri 1-3-4-0/gponolt sec 1-4-4-0/gponolt
```

Protection pair has been created. 1-3-4-0/gponolt is primary and 1-4-4-0/gponolt is secondary.

3 Verify the line redundancy



Note: You should wait until redundancy is confirmed before changing any provisioning on the port. Verify the redundancy using one of the following show commands before adding or deleting bridge interfaces or IP interfaces on the OLT port.

Notice that the **gponolt show redund** command will warn you if an OLT port is not yet configured for redundancy.

```
zSH> gponolt show redund
```

```
ERROR! Failed to collect status for OLT 4/4
```

OLT	Interface	Status	Redundancy State	Redundancy Peer OLT	Redundancy Peer Interface
3/1	1-3-1-0	00S			
3/2	1-3-2-0	00S			
3/3	1-3-3-0	00S			
3/4	1-3-4-0	UP	Primary	4/4	1-4-4-0

OLT	Interface	Status	Redundancy State	Redundancy Peer OLT	Redundancy Peer Interface
4/1	1-4-1-0	00S			
4/2	1-4-2-0	00S			
4/3	1-4-3-0	00S			

When the OLT port is ready, it will be displayed in the **gponolt show redund** command:

zSH> **gponolt show redund**

OLT	Interface	Status	Redundancy State	Redundancy Peer OLT	Redundancy Peer Interface
3/1	1-3-1-0	00S			
3/2	1-3-2-0	00S			
3/3	1-3-3-0	00S			
3/4	1-3-4-0	UP	Primary	4/4	1-4-4-0

OLT	Interface	Status	Redundancy State	Redundancy Peer OLT	Redundancy Peer Interface
4/1	1-4-1-0	00S			
4/2	1-4-2-0	00S			
4/3	1-4-3-0	00S			
4/4	1-4-4-0	Trfc-Disable	Secondary	3/4	1-3-4-0

You can also show the redundancy by the specific line

zSH> **line-red show 1-3-4-0/gponolt**

redundancy status for 1-3-4-0/gponolt:

NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0			
Interface-Type	Interface-Name	Oper-State	Oper-Status
Primary	1-3-4-0/gponolt	Active	UP
Secondary	1-4-4-0/gponolt	Standby	Trfc-Disable

4 Create interfaces on the GPON port

We will create a bridge interface:

```
zSH> bridge add 1-3-4-501/gponport gtp 1 downlink vlan 200 tagged
GEM Port 1-3-4-501/gponport has been created on ONU 1-3-4-1/gpononu.
Adding bridge on 1-3-4-501/gponport
Created bridge-interface-record 1-3-4-501-gponport-200/bridge
```

Show the bridge:

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
upl Tagged	200	ethernet5-200/bridge	UP	S	VLAN 200 default
dwn Tagged	200	1-3-4-501-gponport-200/bridge	UP	D	00:00:86:43:3c:e4

5 Test line redundancy

a Bounce the port

```
zSH> port bounce 1-3-4-0/gponolt
1-3-4-0/gponolt set to admin state DOWN
1-3-4-0/gponolt set to admin state UP
```

b Show the line redundancy

```
zSH> line-red show 1-4-4-0
redundancy status for 1-4-4-0/gponolt:
      NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0
```

Interface-Type	Interface-Name	Oper-State	Oper-Status
Primary	1-3-4-0/gponolt	Standby	Trfc-Disable
Secondary	1-4-4-0/gponolt	Active	UP

c Show the bridge.

Notice that even though the port on card 4 is now the active port the name of the bridge does not change (and makes it look like the bridge is coming from the port on card 3, 1-3-4-501-gponport-200/bridge).

```
zSH> bridge show
```

Type	VLAN	Bridge	St	Table	Data
upl Tagged	200	ethernet5-200/bridge	UP	S	VLAN 200 default
dwn Tagged	200	1-3-4-501-gponport-200/bridge	UP	D	00:00:86:43:3c:e4

d Bounce the other port to get it to return to the initial redundancy state

```
zSH> port bounce 1-4-4-0
1-4-4-0 set to admin state DOWN
1-4-4-0 set to admin state UP
```

e Show the line redundancy

```
zSH> line-red show 1-4-4-0
redundancy status for 1-4-4-0/gponolt:
      NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0
```

Interface-Type	Interface-Name	Oper-State	Oper-Status
Primary	1-3-4-0/gponolt	Active	UP
Secondary	1-4-4-0/gponolt	Standby	Trfc-Disable

f Show the bridge

SH> **bridge show**

Type	VLAN	Bridge	St	Table	Data
upl Tagged	200	ethernet5-200/bridge	UP	S	VLAN 200 default
dwn Tagged	200	1-3-4-501-gponport-200/bridge	UP	D	00:00:86:43:3c:e4

Removing a redundant OLT port

Redundancy may be removed from an OLT port, however there are limitations. The original primary port cannot be removed. Active ports can also not be removed.

To resolve downed ports which are on the primary port, resolve the problem with the port (whether downed link or card issue). Resolving the problem can include replacing the card with a new card, then running the **card add** command. When the new card comes up, the redundancy will be reestablished.

1 Show the current status of the redundancy group

zSH> **line-red show 1-3-4-0/gponolt**
redundancy status for 1-3-4-0/gponolt:
NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0

Interface-Type	Interface-Name	Oper-State	Oper-Status
Primary	1-3-4-0/gponolt	Active	UP
Secondary	1-4-4-0/gponolt	Standby	Trfc-Disable

2 Remove the standby port from the redundancy group

zSH> **line-red remove 1-4-4-0/gponolt**
Interface 1-4-4-0/gponolt is no longer in a protection group.



Note: Notice that you cannot remove the primary port of a redundant pair even if it is in standby mode. You also cannot remove the active port (even if it was initially the secondary port of a redundant port).

3 Show that the redundant port has been removed



Note: You should wait until you confirm that redundancy has been removed before changing any provisioning on the port. Verify the redundancy using one of the following show commands before adding or deleting bridge interfaces or IP interfaces on the OLT port.

```
zSH> line-red show 1-3-4-0/gponolt
redundancy status for 1-3-4-0/gponolt:
      NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0
```

The 1-3-4-0/gponolt is not part of any redundancy group

Switchover from active to standby GPON port

A switchover from active to standby GPON port can be done automatically or forced manually using the **port bounce** command.

Automatically switched

A switchover can be triggered automatically when:

- Loss of signal from all ONUs connected to the active GPON port occurs. This could be caused by:
 - The Fiber between the splitter and MXK is down (i.e.the fiber is cut or pulled)
 - Loss of all ONUs on this GPON port

If one or more ONUs go down with still a few ONUs active, it would not indicate a fiber failure between the splitter and MXK, and hence no action is taken by the SLMS software.
 - Loss or damage of splitter
- An SFP for this GPON port is damaged so it does not pass signal or the SFP is removed
- The GPON card is removed or deleted or the card is rebooted

When a switchover happens automatically, it raises an alarm.

Manually switched

A switchover also can be manually switched by the operator by using the **port bounce interface/physical interface type** command:

```
zSH> port bounce 1-3-4-0/gponolt
1-3-4-0/gponolt set to admin state DOWN
1-3-4-0/gponolt set to admin state UP
```

GPON redundancy configuration limitations

The following limitations apply to GPON redundancy configurations:

- When a standby port is added, the configuration information is automatically inherited. If a port is configured as standby, the user cannot enter configuration on the port.
- The card which has proposed secondary ports must be a running card. Before you can use a newly installed card, you must add the card using the **card add** command.
- You cannot add a secondary OLT port which has any added ONUs/ONTs, whether active or not. The port cannot be provisioned with logical interfaces, whether bridge or IP.

If there are active ONUs/ONTs in a standalone port that is being attempted to added as a standby to a redundancy group, the command is rejected. However, a GPON port with active ONUs/ONTs can be moved into a redundancy group as the primary active port.

- A GPON port may only be a member of one redundancy group.
- A GPON port may only be made redundant with another GPON port.
- GPON redundancy may only be with a GPON port on another GPON line card; it cannot be with another GPON port on the same card.
- The following rules apply to deleting ports from OLT redundancy groups:
 - An active port can never be deleted from the redundancy group. If the active port is the secondary port of the redundancy group, neither port can be removed.
 - Only the secondary port of a redundancy group can be deleted (and only when not active).
 - The primary port can never be deleted from the redundancy group.
- Upgrades cannot be scheduled on standby ports



Note: If a switchover event is triggered when an upgrade is in progress, the upgrade is re-queued for the ONUs/ONTs that were in progress as well as the ONUs/ONTs that were currently queued.

MXK GPON using the Reg ID for provisioning

The registration ID (Reg ID) provides an alternative, hardware-independent, method for pre-provisioning an ONU. It allows service providers a flexible method for provisioning PON circuits.

The Reg ID is a 10-digit number associated with ONU that is entered at the OLT. This Reg ID is also used as an ONU password, as defined in G.984.3.

A related feature, password protection, has also been added. This feature enables you to password protect ONUs. If this is enabled on an ONU, each time the ONU is ranged, the password is requested by the OLT and is checked for a match. The ONU doesn't come up if the retrieved password doesn't match the password provisioned at the OLT.

The Reg ID process is as follows:

- When the OLT discovers a serial number it tries to match it against its provisioned serial numbers.
- If there is a serial number match, then:
 - If auto-learn is disabled, the OLT fetches the ONU password and compares it to its stored one, only continuing if the password matches.
 - If auto-learn is enabled, ONU is brought up without password retrieval.

- If there is no match on serial number, then:

The password is retrieved and compared against the password for each ONU configured with `useRegId = True`. If there is a match, the ONU is assigned the serial number and is brought up.

Configuring Reg ID

The **gpononu set** command enables you to configure the Reg ID and password protection options.

```
zSH> gpononu set <slot/olt/onu> regid <xxx>
```

This command enables provisioning by Reg Id.

It sets the password to "xxx", the **use-reg-id** parameter to **Enabled**, and the **onu-added** parameter to **True** in the **gpon-olt-onu-config** profile.

oIf the field already has a password (Reg ID) and it doesn't match the one in the command, the system will ask you to confirm that you want to change it.



Note: The Reg ID must be unique within the OLT.

```
zSH> gpononu set <slot/olt/onu> password <xxx>
```

This sets the password to "xxx" and **auto-learn** to **Disable**.

If the password exists, and it doesn't match the one in the command, the system will ask you to confirm that you want to change it.

Error! Password does not match regId already in use for ONU.

RegId must be cleared first.

```
zSH> gpononu clear <slot/olt/onu> regid
```

Sets **use-reg-id** to **False**, which disables auto-provisioning by Reg ID.

In addition:

- If **auto-learn** is **Enabled** (that is the parameter is not being used as a password), this command clears the password field to an empty string.
- If there is no serial number, then it sets the onu-added parameter to False.
- If there is a serial number, this command does not erase it or bring down the ONU.

```
zSH> gpononu clear <slot/olt/onu> password
```

Sets auto-learn to enable, so that password checking is no longer performed.

In addition:

- If use-reg-id = Disabled (the parameter is not being used as a Reg ID), this clears the password field to an empty string.
- If there is a serial number, this command does not erase it or bring down the ONU.

```
zSH> gpononu clear <slot/olt/onu>
```

Does not modify the **password**, **auto-learn**, or **use-reg-id** parameters.

If **use-reg-id** is set, does not clear **onu-added**.



Note:

The contents of the password is never displayed - it can only be viewed in the profile.

GPON ONU serial number format (Hexadecimal or Decimal)

By default, GPON ONU serial number displayed in hexadecimal. It could also be displayed in decimal.

Display the available ONUs and the discovered serial numbers in hex format with the **gpononu show slot/olt** command:

```
zSH> gpononu show 5/1
```

Processing list of 64

This command may take several minutes to complete.

Do you want to continue? (yes or no) [no] **yes**

Free ONUs for slot 5 olt 1:

2	3	4	5	6	7	8	9	10	11	12	13
14	15	16	17	18	19	20	21	22	23	24	25
26	27	28	29	30	31	32	33	34	35	36	37
38	39	40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59	60	61
62	63	64									

Discovered serial numbers for slot 5 olt 1:

sernoID	Vendor	Serial Number	sernoID	Vendor	Serial Number

2 ZNTS 00F1B37F 3 ZNTS 84200459

Display the serial number in decimal format with the **gpononu show slot/olt -d** command:

zSH> **gpononu show 5/1 -d**

Processing list of 64

This command may take several minutes to complete.

Do you want to continue? (yes or no) [no] **yes**

Free ONUs for slot 5 olt 1:

2	3	4	5	6	7	8	9	10	11	12	13
14	15	16	17	18	19	20	21	22	23	24	25
26	27	28	29	30	31	32	33	34	35	36	37
38	39	40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59	60	61
62	63	64									

Discovered serial numbers for slot 5 olt 1:

sernoID	Vendor	Serial Number	sernoID	Vendor	Serial Number
2	ZNTS	15840127	3	ZNTS	2216690777

Display the serial number in decimal format with the **gpononu showall slot/olt -d** command:

zSH> **gpononu showall 7/7 -d**

Processing list of 64

This command may take several minutes to complete.

Do you want to continue? (yes or no) [no] **yes**

ONU	Name	Enabled	Model #	Serial Number	OMCI files and profiles
1	1-7-7-1	Yes		ZNTS 1341	ME 2210-me GEN 2210-gen
2	1-7-7-2	Yes		ZNTS 1405	ME 2210-me GEN 2210-gen
3	1-7-7-3	Yes		ZNTS 1263	ME 2210-me GEN 2210-gen
4	1-7-7-4	Yes		ZNTS 1359	ME 2210-me GEN 2210-gen
5	1-7-7-5	Yes		ZNTS 1285	ME 2210-me GEN 2210-gen
6	1-7-7-6	Yes		ZNTS 1387	ME 2210-me GEN 2210-gen
7	1-7-7-7	Yes		ZNTS 1335	ME 2210-me GEN 2210-gen
8	1-7-7-8	Yes		ZNTS 1371	ME 2210-me GEN 2210-gen

<SPACE> for next page, <CR> for next line, A for all, Q to quit

Associate a vendor ID and a serial number with an ONU when activating the ONU

Enable an ONU with the vendor ID and serial number by using the **gpononu set slot/olt/onu vendorid vendorId serno [fsan a hex number] | [a decimal number]** command. You can specify serial number in hex or decimal format. **fsan** indicates the serial number is in hex format.

Usually the vendor ID and serial number can be found in a sticker on the ONU. For example, a small sticker on an ONU 2510 shows the FSAN serial number, e.g. FSAN-ZNTS00F1B37F. The first four characters, ZNTS, are vendor specific ID, and the following characters, 00F1B37F, are serial number in hex format.

Associate a vendor ID and a hex serial number with an ONU and enable this ONU:

```
zSH> gpononu set 5/1/2 vendorid ZNTS serno fsan 00F1B37F
Onu 2 successfully enabled with serial number ZNTS 00F1B37F
```

Associate a vendor ID and a decimal serial number with an ONU and enable this ONU:

```
zSH> gpononu show 5/1 -d
Processing list of 64
This command may take several minutes to complete.
Do you want to continue? (yes or no) [no] yes
Free ONUs for slot 5 olt 1:
   3    4    5    6    7    8    9   10   11   12   13   14
  15   16   17   18   19   20   21   22   23   24   25   26
  27   28   29   30   31   32   33   34   35   36   37   38
  39   40   41   42   43   44   45   46   47   48   49   50
  51   52   53   54   55   56   57   58   59   60   61   62
  63   64
Discovered serial numbers for slot 5 olt 1:
sernoID  Vendor  Serial Number
  3       ZNTS   2216690777
```

```
zSH> gpononu set 5/1/3 vendorid ZNTS serno 2216690777
Onu 3 successfully enabled with serial number ZNTS 2216690777
```

GPON extended reach

The MXK GPON solution supports extended reach. The maximum distance between the MXK and the farthest ONT/ONU is 60 Km. The requirement when deploying ONTs are the maximum distance between two ONTs cannot exceed 20Km. The solution can be deployed by using GPON Class B+ SFP optics supported by Zhone.

A key point to keep in mind when deploying GPON is optical budget. This becomes more critical when the distance between MXK and ONT is higher. Listed below are factors that affect optical budget:

1. Higher the distance, higher the optical loss
2. Greater the number of splits, greater the optical loss
3. Cascading splitters causes higher loss than non-cascaded splits
4. Every connector introduces optical loss
5. Dirty fiber increases optical loss

It is recommended when deploying GPON:

1. Always clean the fiber before deploying
2. Use only fiber cleaning kit to clean fiber and not any other cleaning solution
3. Cap the unused splitters
4. Calculate the optical budget based on distance and fiber
5. Always plan for some margin in optical budget

Recommendations for extended reach

40 Km:

- Maximum Distance between MXK and farthest ONT: 40 Km
- Minimum Distance between MXK and closest ONT to MXK: 20 Km
- Maximum distance between any two ONTs: 20 Km (Note this is always a constant)
- Recommended # of splits: 32

50 Km:

- Maximum Distance between MXK and farthest ONT: 50 Km
- Minimum Distance between MXK and closest ONT to MXK: 30 Km
- Maximum distance between any two ONTs: 20 Km (Note this is always a constant)
- Recommended # of splits: 8

60 Km:

- Maximum Distance between MXK and farthest ONT: 60 Km
- Minimum Distance between MXK and closest ONT to MXK: 40 Km
- Maximum distance between any two ONTs: 20 Km (Note this is always a constant)
- Recommended # of splits: 4

Commands to enable extended reach

By default, the MXK supports a maximum distance of 20 Km between MXK and ONT. The two parameters used to increase the supported distance on GPON are the *max-rt-propagation-delay* and *min-rt-propagation-delay* parameters in the **gpon-olt-config** profile. The default value for these parameters are 200 and 0 respectively. For every 1 Km increase the value of both these parameters have to be increased by a value of 10. So to increase the distance by 10 Km from the default value of 20 Km to 30 Km, change the *max-rt-propagation-delay* parameter in the **gpon-olt-config** profile to 300 and the *min-rt-propagation-delay* parameter in the **gpon-olt-config profile** to 100. Both these parameters have to be modified together as they go together in ensuring all ONTs can communicate with the MXK.

View the values of a profile with the get command:

```
zSH> get gpon-olt-config 1-1-1-0/gponolt

gpon-olt-config 1-1-1-0/gponolt
max-rt-propagation-delay : ----> {200}
max-onu-response-time: -----> {50}
preassigned-eqd: -----> {0}
los-alpha: -----> {4}
lof-alpha: -----> {4}
loam-alpha: -----> {3}
scrambler: -----> {enabled}
fec-mode: -----> {disabled}
auto-learn: -----> {enabled}
power-level: -----> {0}
guard-bit-count: -----> {32}
dba-mode: -----> {predictive}
gem-block-size: -----> {16}
us-ber-interval: -----> {5000}
ds-ber-interval: -----> {5000}
ber-sf-threshold: -----> {3}
ber-sd-threshold: -----> {5}
fec-request: -----> {disabled}
key-exchange: -----> {disabled}
min-rt-propagation-delay : ----> {0}
min-onu-response-time: -----> {10}
eqd-measure-cycles: -----> {5}
drift-ctrl-interval: -----> {1000}
drift-ctrl-limit: -----> {3}
alloc-cycle-length: -----> {2}
min-us-alloc: -----> {16}
ack-timeout: -----> {2000}
pls-max-alloc-size: -----> {120}
dba-cycle: -----> {2}
sr-dba-reporting-block-size: -> {48}
```

Update the profile with the update command:

```
zSH> update gpon-olt-config 1-1-1-0/gponolt
```

```

gpon-olt-config 1-1-1-0/gponolt
max-rt-propagation-delay : ----> {200} 300
max-onu-response-time: -----> {50}
preassigned-eqd: -----> {0}
los-alpha: -----> {4}
lof-alpha: -----> {4}
loam-alpha: -----> {3}
scrambler: -----> {enabled}
fec-mode: -----> {disabled}
auto-learn: -----> {enabled}
power-level: -----> {0}
guard-bit-count: -----> {32}
dba-mode: -----> {predictive}
gem-block-size: -----> {16}
us-ber-interval: -----> {5000}
ds-ber-interval: -----> {5000}
ber-sf-threshold: -----> {3}
ber-sd-threshold: -----> {5}
fec-request: -----> {disabled}
key-exchange: -----> {disabled}
min-rt-propagation-delay : ----> {0} 100
min-onu-response-time: -----> {10}
eqd-measure-cycles: -----> {5}
drift-ctrl-interval: -----> {1000}
drift-ctrl-limit: -----> {3}
alloc-cycle-length: -----> {2}
min-us-alloc: -----> {16}
ack-timeout: -----> {2000}
pls-max-alloc-size: -----> {120}
dba-cycle: -----> {2}
sr-dba-reporting-block-size: -> {48}
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Received Signal Strength Indication (RSSI) and Digital Diagnostic Monitoring (DDM)

Viewing the OLT and ONU optical power

Received Signal Strength Indication (RSSI) is the capability of a SFP by which the SFP reads the strength of the signal received on the OLT from an ONU. The MXK-GPON-SFP-B+-RSSI supports RSSI feature.

To display the optical power received on both OLT and ONU side, use the **gpononu power show** [slot [/olt[/onu]]]command.

Note that the *ONT receive power* should be -28 or above for SFP-B+. Otherwise it is recommended to remove some splitters between the ONU and the connected OLT.

The following example shows the ONT receive power on 1/7/3/1 is -23.0 dBm, in the normal range.

```
zSH> gpononu power show 7/3
Processing list of 64
This command may take several minutes to complete.
Do you want to continue? (yes or no) [no] yes
```

Interface	OLT Receive Power	ONT Receive Power
=====	=====	=====
1-7-3-1	-17.4 dBm	-23.0 dBm
1-7-3-2	-17.2 dBm	-23.0 dBm
1-7-3-3	-18.2 dBm	-23.0 dBm
1-7-3-4	-18.0 dBm	-23.0 dBm
1-7-3-5	-17.4 dBm	-23.0 dBm
1-7-3-6	-17.8 dBm	-24.0 dBm
1-7-3-7	-16.9 dBm	-23.0 dBm
1-7-3-8	-17.4 dBm	-23.0 dBm
1-7-3-9	-17.3 dBm	-23.0 dBm
1-7-3-10	-17.2 dBm	-22.0 dBm
1-7-3-11 error	-25.0	dBm
1-7-3-12	NA	NA

<SPACE> for next page, <CR> for next line, A for all, Q to quit **q**

If there is no SFP inserted in the OLT, or the OLT/ ONU admin status is set to down, then its Receive Power field displays the value “NA”.

If the Receive Power field displays the value “error“, it means the measurement failed. You can run the **gpononu power show** command again.

Viewing the transmit parameters on OLT

Digital Diagnostic Monitoring (DDM) provides diagnostic information about the SFP. With DDM function, the SFP optical transceiver, MXK-GPON-SFP-B+-RSSI, measures the transceiver temperature, transceiver supply voltage, Tx Bias current, and Tx output power parameters on an OLT, and also reports an End of Life status. An alarm is raised when an End of Life condition is reached.

Perform DDM on the GPON OLT card with the **gponolt show port** [slot [/olt]] command.

[Table20](#) provides the output fields description for this command.

Table 20: Gponolt port show Command Output Field Explanations

Field	Description
SLOT/OLT	The GPON ID.
Temperature	Internally measured Transceiver Temperature of the OLT in Celsius.

Table 20: Gponolt port show Command Output Field Explanations

Field	Description
Voltage	Internally measured Transceiver Supply Voltage of the OLT in Volts.
Tx Bias Current	Measured Tx Bias current per OLT in Milli Amperes.
Tx Power	Measured Tx Output Power of the OLT in dBm.
End of Life Status	When the End Of Life (EOL) Alarm bit is set an alarm will be raised. SFP automatically maintains a laser output optical power by adjusting the laser current. Alarm is raised when the SFP reaches the end of life which is about 150% of original current. Alarm will be cleared when the SFP is connected. The alarm severity level is Major. Values: ok No alarm conditions are raised warning Warning is set when EOL is at about 130% original current. alarm Alarm conditions are raised SFP not present SFP is not detected

zSH> gponolt show port

```

SLOT/OLT Temperature  Voltage Tx Bias Current Tx Power End Of Life Status
=====
7/1      NA      NA      NA      NA      SFP not present
7/2      NA      NA      NA      NA      SFP not present
7/3      43c     3.3v     10mA     3.0dBm0k
7/4      NA      NA      NA      NA      SFP not present
7/5      NA      NA      NA      NA      SFP not present
7/6      NA      NA      NA      NA      SFP not present
7/7      NA      NA      NA      NA      SFP not present
7/8      NA      NA      NA      NA      SFP not present
SLOT/OLT Temperature  Voltage Tx Bias Current Tx Power End Of Life Status
=====
11/1     46c     2.9v     22mA     2.1dBm0k
11/2     NA      NA      NA      NA      SFP not present
11/3     NA      NA      NA      NA      SFP not present
11/4     35c     3.3v     25mA     4.1dBm0k

```

Manage ONU with OMCI

This section describes how GPON ONUs are managed with OMCI (standard G984.1 - 984.4):

Topics:

- [View OMCI configuration states and alarms on ONU, page434](#)
- [Retrieve status of GPON subscriber ports, page436](#)
- [Administration of GPON subscriber ports, page436](#)
- [Configurable speed of GPON subscriber ports, page437](#)
- [Configurable Transmit Gain, Receive Gain, and Fax Mode parameters for POTS ports, page437](#)
- [Manual upgrade on an ONU, page439](#)
- [Auto upgrade on an ONU, page443](#)
- [View the ONU upgrade status, page447](#)
- [Reboot an ONU, page449](#)
- [Re-synchronize an ONU, page449](#)
- [Add new services for subscriber without affecting existing services, page449](#)
- [Retrieve alarm information on an ONU, page449](#)
- [View or change trap reporting status on an ONU, page450](#)

View OMCI configuration states and alarms on ONU

View OMCI configuration states and alarms generated on an ONU with the `gpononu status` command.

[Table21](#) provides the output fields description for this command.

Table 21: Gpononu status Command Output Field Explanations

Field	Description
ID	The ONU ID. In the range of 1 to 64.
Onu	The ONU interface name. By default in the format of shelf ID-Slot ID-OLT ID-ONU ID
OperStatus	The operational status of the ONU. Values: Up Down

Table 21: Gpononu status Command Output Field Explanations

Field	Description
OmciConfigState	The OMCI configuration states on the ONU. It is detected by the OLT side with respect to the ONU. Values: Waiting The ONU is queued to be configured, while other ONUs are being handled. Pending The ONU is being processed but not yet being configured. For profile-based provisioning, this means the OMCI profiles are being read and prepared. For file-based provisioning, the OMCI file is being downloaded. Config Configuration is in progress. Error An error occurred during configuration, or the profiles could not be found, etc. Done Successfully configured.
GponOnuStatus	The standard GPON MAC alarms of the ONU detected on the OLT. Values: Active ONU is active, no alarm Inactive ONU is inactive, cannot get alarm LOS Lost of Signal LOF Lost of Frame DOW Drift of Window DG Dying Gasp SF Signal Fail SD Signal Degrade LCDG Lost of GEM Channel Delinquency RD Remote Defect TF Transmitter Failure SUF Start Up Failure LOA Lost of Ack MEM Message error PEE Physical equipment error OAML Lost of OAM

This example shows an ONU that is enabled and completes the OMCI provisioning.

```
zSH> gpononu status 13/1/1
```

```
Slot 13 olt 1
ID      Onu      OperStatus  OmciConfigState  GponOnuStatus
===  =====  =====  =====  =====
1      1-13-1-1    Up        Done        Active
```

This example shows an ONU is enabled and then goes down with a dying gasp.

```
zSH> gpononu status 18/7/1
```

Slot 18	olt 7			
ID	Onu	OperStatus	OmciConfigState	GponOnuStatus
1	1-18-7-1	Down	Done	Inactive+LOS+LOF+DG+OAML

Retrieve status of GPON subscriber ports

From the MXK, the administrative state and operational state of subscriber facing ports on ONU can be retrieved by using this command:

```
gpononu status [slot[/olt[/onu]] | interfaceName] port all[portType
[portNumber]
```

The portType is based on what are supported by the ONU model. The possible port types could be *eth* (ethernet port), *pots* (POTS port), *rf* (RF port), and *ces* (T1/E1 port).

This example shows the status of an ethernet port on ONU:

```
zSH> gpononu status 5/1/1 port eth 1
5/1/1 ONU Port Status
      Ethernet Port Status - Port 1
      Administrative State      up
      Operational State        active
```

Administration of GPON subscriber ports

Enabling or disabling admin state of a GPON subscriber port

From the MXK, the administrative state of an ONU subscriber facing port can be enabled or disabled by using this command:

```
gpononu port [slot[/olt[/onu]] portType interfaceID up|down
```

The portType is based on what are supported by the ONU model. The possible port types could be *eth* (ethernet port), *pots* (POTS port), *rf* (RF port), and *ces* (T1/E1 port).

- 1 Set an ONU ethernet port admin state to down:

```
zSH> gpononu port 5/1/1 eth 1 down
```

- 2 Verify the admin status:

```
zSH> gpononu status 5/1/1 port eth 1
5/1/1 ONU Port Status
      Ethernet Port Status - Port 1
      Administrative State      down
      Operational State        inactive
```

- 3 Set the ONU ethernet port admin state back to up:

```
zSH> gpononu port 5/1/1 eth 1 up
```

- 4 Verify the admin state is up:

```
zSh> gpononu status 5/1/1 port eth 1
5/1/1 ONU Port Status
      Ethernet Port Status - Port 1
      Administrative State      up
      Operational State        active
```

Configurable speed of GPON subscriber ports

By default the GPON ONU port speed of subscriber facing ports is set to auto-detect. You can modify this setting by using the **gpononu auto-detect** command:

```
zSh> gpononu auto-detect<slot>/<olt>/<onu> portType <interface#>
      < auto | 10F | 100F | 1000F | 10H | 100H | 1000H | 10FA | 1000A >
```

The settings for this command are:

- auto: auto-detect
- 10F: 10 Mbps, full duplex only
- 100F: 100 Mbps, full duplex only
- 1000F: 1000 Mbps, full duplex only
- 10H: 10 Mbps, half duplex only
- 100H: 100 Mbps, half duplex only
- 1000H: 1000 Mbps, half duplex only
- 10FA: 10 Mbps full duplex and auto
- 1000A: 1000 Mbps auto

The following example configures an ONU Ethernet subscriber port for 1000 full duplex mode:

```
zSh> gpononu auto-detect 10/1/1 eth 1 1000F
```

Configurable Transmit Gain, Receive Gain, and Fax Mode parameters for POTS ports

The OMCI Transmit Gain, Receive Gain, and Fax Mode parameters can be set using the Smart OMCI Configuration Utility tool on the Zhone website (<http://www.zhone.com/support/tools/omci>) or by using the **gpononu profiles update gen** command.

The **Rx Gain** and **Tx Gain** parameters configure the sensitivity of POTS ports for gain and attenuation. The Fax mode parameter defines the G.711 or T.38:

- Rx Gain

Specifies the gain value for the received signal in the form of a 2s complement number. Valid values are -120 (-12.0 dB) to 60 (+6.0 dB). The default value is 0.

- Tx Gain

Specifies the gain value for the transmit signal in the form of a 2s complement number. Valid values are -120 (-12.0 dB) to 60 (+6.0 dB). The default value is 0.

- Fax mode.

0 Passthru

1 T.38

The default value is 0 (Passthru).

Assign values to these parameters in the Generic profile. Use the **gpononu profiles update gen** command, then enter the corresponding variable indexes and values.

The following example shows how to configure these parameters.

```

ZSH> gpononu profiles show gen
2520-GEN
ZSH> gpononu profiles update gen 2520-GEN
Generic Profile: 2520-GEN
 1 "ETH1 Tagging mode: 0-Pass through, 1-Tagging, 2-QinQ []" 1
 2 "ETH1 VLAN (VID or COS,VID) [0,]" 1
 3 "ETH1 Forward Oper []" 0xf
 4 "Eth 1 Auto Detection [0]" 0
 5 "ETH2 Tagging mode: 0-Pass through, 1-Tagging, 2-QinQ []" 1
 6 "ETH2 VLAN (VID or COS,VID) [0,]" 1
 7 "ETH2 Forward Oper []" 0xf
 8 "Eth 2 Auto Detection [0]" 0
 9 "ETH3 Tagging mode: 0-Pass through, 1-Tagging, 2-QinQ []" 1
10 "ETH3 VLAN (VID or COS,VID) [0,]" 1
11 "ETH3 Forward Oper []" 0xf
12 "Eth 3 Auto Detection [0]" 0
13 "ETH4 Tagging mode: 0-Pass through, 1-Tagging, 2-QinQ []" 1
14 "ETH4 VLAN (VID or COS,VID) [0,]" 1
15 "ETH4 Forward Oper []" 0xf
16 "Eth 4 Auto Detection [0]" 0
17 "Voice VLAN [0,5]" 5
18 "VOIP Host IP Option: 2-static, 3-DHCP [3]" 3
19 "VOIP Host IP [0.0.0.0]"
20 "VOIP Netmask [0.0.0.0]"
21 "VOIP Gateway [0.0.0.0]"
22 "VOIP Server IP [172.16.60.51]"
23 "VOIP Primary DNS [172.16.1.5]" 172.16.1.5
24 "VOIP Secondary DNS [172.16.1.10]" 172.16.1.10
25 "Country Code [ 1]"
26 "Signalling Code [1]"
27 "Impedance [2]"
28 "Rx Gain [0]" 0
29 "Tx Gain [0]" 0
30 "Out-of-band DTMF [0]"

```

```

31 "POTS1 Dial Number []" 2012020011
32 "POTS1 User Name []" 2012020011
33 "POTS1 Password []" 123456
34 "POTS2 Dial Number []" 2012020012
35 "POTS2 User Name []" 2012020012
36 "POTS2 Password []" 123456
37 "POTS3 Dial Number []" 2012020013
38 "POTS3 User Name []" 2012020013
39 "POTS3 Password []" 123456
40 "POTS4 Dial Number []" 2012020014
41 "POTS4 User Name []" 2012020014
42 "POTS4 Password []" 123456
43 "Announcement Type [0xFF]"
44 "Echo Cancel: 1-enable, 0-disable [1]"
45 "Fax Mode [0]"
46 "Dial Plan Format [1]"
47 "CID Features [45]"
48 "Call Waiting Features [3]"
49 "Call Progress or Transfer Features [165]"
50 "Call Present Features [15]"
51 "SIP Domain Part 1 [metaswitch.oak.zhong.com]" metaswitch.oak.zhong.com
52 "SipRegistrar [172.16.60.51]" metaswitch.oak.zhong.com
53 "video Admin Status: 0-Up, 1-Down [0]" 0
54 "video Admin Status: 0-Up, 1-Down [0]" 0
55 "video Admin Status: 0-Up, 1-Down [0]" 0
56 "video Admin Status: 0-Up, 1-Down [0]" 0
57 "POTS 1 Admin Status: 0-Up, 1-Down [0]" 0
58 "POTS 2 Admin Status: 0-Up, 1-Down [0]" 0
59 "POTS 3 Admin Status: 0-Up, 1-Down [0]" 0
60 "POTS 4 Admin Status: 0-Up, 1-Down [0]" 0
Enter OMCI edit command or [s]ave, [q]uit, [h]elp: 28
"Rx Gain [0]" 0: 12
Enter OMCI edit command or [s]ave, [q]uit, [h]elp: 29
"Tx Gain [0]" 0: 12
Enter OMCI edit command or [s]ave, [q]uit, [h]elp: 45
"Fax Mode [0]" : 1
Enter OMCI edit command or [s]ave, [q]uit, [h]elp: s
GENERIC profile has been saved

```

Manual upgrade on an ONU

The OMCI image upgrade feature enables MXK user to manually upgrade firmware on an ONU immediately. Before downloading the image file to the ONU, make sure the image file exists in the MXK first. The OMCI standard defines two managed entities for firmware image, which are referred to as partitions 0 and partition 1.

These actions are supported on OMCI image manual upgrade feature:

- Download
Download an image file from OLT to ONU.
- Activate

Reboot using an image in the specified partition.

- Commit

Set the image in the specified partition as the boot default.

- Abort

Terminate the queued download.

According to the OMCI standard, download is only allowed if the partition is neither active nor committed. Similarly, activate and commit are only allowed if the partition is valid.

After successfully downloading an image file, ONU automatically checks whether the image file is valid. Only valid file could be able to activated.

Upgrade firmware image on an ONU from MXK with the **gpononu image** command.

gpononu image *slot/olt/onu* **download** *filename* | **activate** | **commit** | **download-activate** *filename* | **download-activate-commit** *filename* | **abort** | **show** [*part partition#*]

[Table22](#) provides the description for command options in the **gpononu image** command.

Table 22: Gpononu image Command Option Explanations

Command Option	Description
download <i>filename</i> [part <i>partition#</i>]	Download an image file to the ONU from OLT. Part partition number is optional. An image file will be downloaded to either an inactive partition or an uncommitted partition. After downloading, ONU validates the file.
activate [part <i>partition#</i>]	Bootup a valid file in the inactive partition immediately in ONU. Part partition number is optional. Only one partition at a time could be active.
commit [part <i>partition#</i>]	Specify a default file to bootup when next time this ONU is powered up. Part partition number is optional. It will commit the file in the uncommitted partition. Only one partition at a time could be committed.
download-activate <i>filename</i> [part <i>partition#</i>]	Perform the download action, and then if the file passed validation check, perform the activate action. Part partition number is optional.
download-activate-commit <i>filename</i> [part <i>partition#</i>]	Perform the download and activate actions, and then if the ONU ranges, perform the commit action. If ranging doesn't occur within a timeout period, return error. Part partition number is optional.
abort	Terminate the queued download.
show	Show the settings for the files downloaded. You can view the file version, the validation status, the activation status, and the commitment status for each partition. It also provide download status, ONU model ID, Upgrade start time, will be activated or not, will be committed or not, and upgrade type.

Table 22: Gpononu image Command Option Explanations

Command Option	Description
part <i>partition#</i>	You can have two image files stored in the ONU. One in partition 0, one in partition 1.

The following examples describe some common configurations during GPON ONU image upgrading:

- 1 Download image file *2510/2510.img* from MXK OLT to ONU 11/4/24, and activate-commit it at once with the **gpononu image slot/olt/onu download-active-commit** command.

Note that when downloading the image file to the ONU, the ONU image file must exist in the MXK flash. Not specifying the directory in the *fileName* field indicates the file is stored under the root directory, */card1*. And this example didn't specify the partition number, so the file will be downloaded to the current standby partition, partition 0.

```
zSH> gpononu image 11/4/24 download-activate-commit
2510/2510.img
Image "/card1/2510/2510.img" downloading to onu 11/4/
24.
```

- 2 View the upgrade status on this ONU with the **gpononu image slot/olt/onu show** command.

This example shows the image download has been requested, and has been queued by the system for download. The download status is Queued.

```
zSH> gpononu image 11/4/24 show
          Partition 0          Partition 1
-----
Version:      R2.0.8.17      R2.0.8.17
isCommitted:  False         True
isActive:    False         True
isValid:      True          True
Download status: Queued
Onu model id: 2510
Upgrade start time: OCT 01 23:15:32 2009
Will be activated: True
Will be committed: True
Upgrade type: Manual
```

This example shows the image download is in progress or validating the downloaded image is in progress. The download status is Downloading, and the *isValid* status in Partition 0 is False.

```
zSH> gpononu image 11/4/24 show
          Partition 0          Partition 1
-----
Version:      R2.0.8.17      R2.0.8.17
isCommitted:  False         True
isActive:    False         True
```

```

isValid:          False                               True
Download status:  Downloading
Onu model id:     2510
Upgrade start time: OCT 01 23:15:32 2009
Will be activated: True
Will be committed: True
Upgrade type:     Manual

```

This example shows the image file has been downloaded to the ONU and passed validation, but not activated yet. The download status is Downloaded, and isValid status in Partition 0 is True.

```

zSH> gpononu image 11/4/24 show
                                Partition 0          Partition 1
-----
Version:          R2.0.8.17              R2.0.8.17
isCommitted:      False                   True
isActive:         False                   False
isValid:          True                     True
Download status:  Downloaded
Onu model id:     2510
Upgrade start time: OCT 01 23:15:32 2009
Will be activated: False
Will be committed: True
Upgrade type:     Manual

```

This example shows the image file has been activated. The isActive status is True.

```

zSH> gpononu image 11/4/24 show
                                Partition 0          Partition 1
-----
Version:          R2.0.8.17              R2.0.8.17
isCommitted:      False                   True
isActive:         True                   False
isValid:          True                     True
Download status:  Downloaded
Onu model id:     2510
Upgrade start time: OCT 01 23:15:32 2009
Will be activated: False
Will be committed: True
Upgrade type:     Manual

```

This example shows the whole downloading, activating, and committing the image file to the ONU is successfully completed. The isCommitted status is True, and the Download status is Complete.

```

zSH> gpononu image 11/4/24 show
                                Partition 0          Partition 1
-----
Version:          R2.0.8.17              R2.0.8.17
isCommitted:      True                   False
isActive:         True                    False
isValid:          True                     True
Download status:  Complete

```

```

Onu model id:          2510
Upgrade start time:    OCT 01 23:15:32 2009
Will be activated:     False
Will be committed:     False
Upgrade type:          Manual

```

The possible values of the Download status are:

- None: ONU is not running, or not OMCI-provisioned.
- No Upgrade: ONU is running, but hasn't been upgraded yet.
- Queued: An Image download has been requested, and has been queued by the system for download.
- Downloading: Downloading the image to the ONU is in progress or validating the downloaded image is in progress.
- Downloaded: The software has been downloaded to the ONU but not activated yet. One possible reason is the ONU is rebooting.
- Complete: Successfully completed downloading, activating, and committing the software to the ONU.
- Error: Failed to upgrade due to some errors.
- Aborted: The ONU queued to be upgraded was aborted by request.
- FileErr: The software file to be upgraded does not exist, or has errors

Auto upgrade on an ONU

The OMCI ONU auto upgrade feature allows MXK user to automatically upgrade ONUs which are installed with the outdated software images when they are ranging.

If the ONU auto-upgrade is enabled, when an ONU ranges, MXK searches a auto-upgrade template for this ONU model. The auto-upgrade template is pre-defined by the user for each ONU model. It contains the auto-upgrade enable status, model ID, allowed software version, and the image file to be downloaded. If the matching template is found for the specific ONU model, the MXK compares the ONU software version with the allowed software version defined in the template. If they are same, then the auto-upgrade is interrupted, otherwise the MXK automatically upgrades the ONU.

The actions to automatically upgrade an ONU software through OMCI are Download->Activate->Commit.

The download is always performed on the standby partition. If the download is successful, then the standby partition is made the active and then the image is committed to the partition. After the image is committed, the auto-upgrade is finished.

Auto upgrading an ONU

- 1 Create an auto-upgrade template for an ONU model.

a Create an auto-upgrade template:

Note that when creating the template, the image file must already exist in the flash, otherwise an error message displays. Not specifying the directory in the *fileName* field indicates the file is stored under the root directory, */card1*.

```
zSH> new remote-sw-upgrade-profile 4
remote-sw-upgrade-profile 4
Please provide the following: [q]uit.
enabled: ---> {true}:
model: ----> {}: 2510
swVersion: -> (): R2.0.8.17
filename: --> (): zh.sip.cimg.2.0.8.17
.....
Save new record?[s]ave, [c]hange or [q]uit: s
```

b Verify the template is created:

```
zSH> list remote-sw-upgrade-profile
remote-sw-upgrade-profile 4
1 entry found.
```

2 Verify ONU auto-upgrade is enabled:

To perform ONU auto upgrade, both the ONU model auto-upgrade status and ONU ID auto-upgrade status should be enabled. By default, those two places are enabled.

a Verify the auto-upgrade status on an ONU model is enabled in the auto-upgrade template:

```
zSH> update remote-sw-upgrade-profile 4
remote-sw-upgrade-profile 4
Please provide the following: [q]uit.
enabled: ---> {true}:
model: ----> {2510}:
swVersion: -> (R2.0.8.17):
filename: --> (zh.sip.cimg.2.0.8.17):
.....
Save new record?[s]ave, [c]hange or [q]uit: q
```

b Verify the auto-upgrade status on an ONU ID is enabled in the gpon-olt-onu-config profile:

```
zSH> get gpon-olt-onu-config 1-15-1-1/gpononu
gpon-olt-onu-config 1-15-1-1/gpononu
serial-no-vendor-id: -----> {ZNTS}
serial-no-vendor-specific: -----> {0}
password: -----> {}
auto-learn: -----> {enabled}
power-level: -----> {0}
us-ber-interval: -----> {5000}
ds-ber-interval: -----> {5000}
onu-added: -----> {false}
```

```

omci-file-name: -----> {}
ONU-Managed-Entity-Profile-name: -----> {}
ONU-Generic-Assignments-Profile-name: -> {}
physical-traps: -----> {disabled}
ont-traps: -----> {disabled}
line-status-traps: -----> {disabled}
auto-upgrade: -----> {enabled}
serial-no-vendor-specific-fsan: -----> {0}
use-reg-id: -----> {disabled}

```

Or you can view it with the **gpononu auto-upgrade show slot [/olt/[onu]] | all** command:

```

zSH> gpononu auto-upgrade show 15
Processing list of 512
This command may take several minutes to complete.
Do you want to continue? (yes or no) [no] yes
Slot 15 olt 1
ONU      Name      Auto-upgrade
=== =====
  1      1-15-1-1    enabled
  2      1-15-1-2    enabled
  3      1-15-1-3    enabled
  4      1-15-1-4    enabled
  5      1-15-1-5    enabled
<SPACE> for next page, <CR> for next line, A for
all, Q to quit

```

- c** In case the auto-upgrade status on this ONU ID is disabled, it can be enabled by the **gpononu auto-upgrade enable slot [/olt/[onu]] | all** command:

```

zSH> gpononu auto-upgrade enable 15/1
This command may affect many OLTs and ONUs.
Do you want to continue? (yes or no) [no] yes

processing 1-15-1-1/gpononu ...
processing 1-15-1-2/gpononu ...
processing 1-15-1-3/gpononu ...
processing 1-15-1-4/gpononu ...
processing 1-15-1-5/gpononu ...
...
Operation successful.

```

3 Disable ONU auto-upgrade.

Disabling auto-upgrade in the auto-upgrade template or on the individual ONU causes the MXK to abort the queued download.

- a** Disable auto-upgrade for an ONU model in the auto-upgrade template:

```

zSH> update remote-sw-upgrade-profile 4
remote-sw-upgrade-profile 4
Please provide the following: [q]uit.

```

```

enabled: ---> {true}: false
model: ----> {2510}:
swVersion: -> (R2.0.8.17):
filename: --> (zh.sip.cimg.2.0.8.17):
.....
Save new record?[s]ave, [c]hange or [q]uit: s

```

- b** Disable auto-upgrade on an ONU with the **gpononu auto-upgrade disable slot [/olt[/onu]] | all** command:

```

zSH> gpononu auto-upgrade disable 15/1
This command may affect many OLTs and ONUs.
Do you want to continue? (yes or no) [no] yes

processing 1-15-1-1/gpononu ...
processing 1-15-1-2/gpononu ...
processing 1-15-1-3/gpononu ...
processing 1-15-1-4/gpononu ...
processing 1-15-1-5/gpononu ...
...
Operation successful.

```

View the ONU upgrade status

- The ONU upgrade status for both auto-upgrade and manual upgrade per ONU can be viewed with the `gpononu upgrade show` command.

This command shows the ONU upgrade state (note that it is same as the Download status for the `gpononu image` command), ONU model ID, the date-time when last upgrade was started, active state, commit state, type of upgrade (Manl or Auto), and which partition is used.

- The detail partition information on each ONU can be viewed with the `gpononu image slot/olt/onu show` command.

Viewing ONU upgrade status

Here are the examples for each show command:

- View the upgrade status on an ONU:

```
zSH> gpononu upgrade show 11/4/2
```

ONU	State	Model	Start Time	Act	Cmt	Typ	Part
11/4/2	Complete	2510	OCT 01 19:37:39 2009	F	F	Auto	1

The possible values of the State field are:

- None: No download requested since slot boot.
- Noupgrade: ONU is running, but hasn't been upgraded yet.
- Queued: An Image download has been requested, and has been queued by the system for download.
- Downloading: Downloading the image to the ONU is in progress or validating the downloaded image is in progress.
- Downloaded: The software has been downloaded to the ONU but not activated yet. One probably reason is the ONU is rebooting.
- Complete: Successfully completed downloading, activating, and committing the software to the ONU.
- Error: Failed to upgrade due to some errors.
- Aborted: The ONU queued to be upgraded was aborted.
- FileErr: The software file to be upgraded does not exist, or has errors

- View upgrade status of the ONUs that have the same upgrade state:

```
zSH> gpononu upgrade show complete
```

ONU	State	Model	Start Time	Act	Cmt	Typ	Part
11/4/1	Complete	2510	OCT 01 22:28:40 2009	F	F	Manl	1
11/4/2	Complete	2510	OCT 01 19:37:39 2009	F	F	Auto	1

- View the upgrade status and the partition information on an ONU:

zSH> **gpononu image 11/4/2 show**

	Partition 0	Partition 1
Version:	R2.0.8.17	R2.0.8.17
isCommitted:	False	True
isActive:	False	True
isValid:	True	True
Download status:	Complete	
Onu model id:	2510	
Upgrade start time:	OCT 01 19:37:39 2009	
Will be activated:	False	
Will be committed:	False	
Upgrade type:	Auto	

4 View all the ONU upgrade status on an OLT:

zSH> **gpononu upgrade show 11/4**

ONU	State	Model	Start Time	Act	Cmt	Typ	P
11/4/11	NoUpgrade	2510					
11/4/12	NoUpgrade	2510					
11/4/13	NoUpgrade	2510					
11/4/14	NoUpgrade	2510					
11/4/15	NoUpgrade	2510					
11/4/16	NoUpgrade	2510					
11/4/17	NoUpgrade	2510					
11/4/18	NoUpgrade	2510					
11/4/19	NoUpgrade	2510					
11/4/20	NoUpgrade	2510					
11/4/21	NoUpgrade	2510					
11/4/22	NoUpgrade	2510					
11/4/23	NoUpgrade	2510					
11/4/25	NoUpgrade	2510					
11/4/26	NoUpgrade	2510					
11/4/27	NoUpgrade	2510					
11/4/28	NoUpgrade	2510					
11/4/29	NoUpgrade	2510					
11/4/30	NoUpgrade	2510					
11/4/31	NoUpgrade	2510					
11/4/32	NoUpgrade	2510					
11/4/1	Complete	2510	OCT 01 22:28:40 2009	F	F	Manl	1
11/4/2	Complete	2510	OCT 01 19:37:39 2009	F	F	Auto	1
11/4/3	Aborted	2510	OCT 01 19:41:37 2009	F	F	Auto	1
11/4/4	Aborted	2510	OCT 01 19:41:38 2009	F	F	Auto	1
11/4/5	Aborted	2510	OCT 01 19:41:42 2009	F	F	Auto	1
11/4/6	Aborted	2510	OCT 01 19:41:46 2009	F	F	Auto	1
11/4/7	Aborted	2510	OCT 01 19:37:59 2009	F	F	Auto	1
11/4/8	Aborted	2510	OCT 01 19:38:00 2009	F	F	Auto	1
11/4/9	Aborted	2510	OCT 01 19:38:05 2009	F	F	Auto	0
11/4/10	Aborted	2510	OCT 01 19:38:10 2009	F	F	Auto	0
11/4/24	Downloaded	2510	OCT 01 23:41:57 2009	F	F	Manl	1

Reboot an ONU

Reboot the remote ONU from MXK with the `gpononu reboot` command.

Reboot an ONU:

```
zSH> gpononu reboot 13/4/2
```

Re-synchronize an ONU

Synchronize an ONU with the MXK with the `gpononu resync` command. This command causes the MXK to break and re-establish linkage with ONU, and send the latest OMCI commands to ONU. It could be used after an ME profile change.

Re-sync an ONU:

```
zSH> gpononu resync 13/4/2
```

Add new services for subscriber without affecting existing services

Adding new services to an existing subscriber does not require a resync of the ONU. For example, if a user has High Speed Internet Access (HSIA) on Ethernet port 1, and requests HSIA on Ethernet port 2, this can be achieved without doing a resync of the ONU by using the `gpononu apply` command.

User can modify the Specific profile or Generic profile and issue the following command:

```
zSH> gpononu apply 13/4/2
```

The `gpononu apply` command issues the OMCI configuration command in the ME profile. This command does not force a resync of the ONU. If user makes modifications to the Specific profile and adds new services, then these commands take effect in the ONU without affecting other services on the same or other ports.

Retrieve alarm information on an ONU

View alarms that are internal to the ONU and ONU LAN facing port with the `gpononu alarms` command.

```
zSH> gpononu alarms 13/4/2
13/4/2 ONU Active Alarms
      PtpEthUni 0x0402 LanLos
      PtpEthUni 0x0404 LanLos
```

View or change trap reporting status on an ONU

The conditions that cause asynchronous reporting traps can be controlled from the OLT through SNMP. The purpose of these controls is to reduce trap traffic between the MXK and ZMS to allow more information about critical or failing ONUs.

These three trap types are reported on an ONU:

- **phy (PhysicalTraps):** Includes the power status, battery status, and physical intrusion conditions as reported from the ONU through OMCI.

The options for the PhysicalTraps are:

- **enable:** The PhysicalTraps are sent.
- **disable:** The PhysicalTraps are not sent. Default value.

- **ont (OntTraps):** The status of LAN facing ports on the ONU (e.g. ethernet port LanLos).

The options for the OntTraps are:

- **enable:** The OntTraps are sent.
- **disable:** The OntTraps are not sent. Default value.

- **line (LineStatusTraps):** It is originated on the MXK, and reports the ONU line going up or down.

The options for the LineStatusTraps are:

- **enable:** The linkUp, linkDown, and lineStatusChange traps are sent.
- **disable:** The lineStatusTraps are not sent. Default value.
- **auto:** In this setting, the linkUp or linkDown traps are not sent, only the lineStatusChange trap is sent if the line is going down with dying gasp (presumably powered down), or if the line is coming up (which may or may not be clearing a dying gasp condition).

View the current reporting status of traps on ONU(s) with the **gpononu traps show [slot[/olt[/onu]]** command.

```
zSH> gpononu traps show 13/4/2
Slot 13 olt 4
ONU      Name      PhysicalTraps  OntTraps  LineStatusTraps
===      =====
2        1-13-4-2      enabled       disabled  auto
```

Change the current reporting status of traps on ONU 13/4/2 with the **gpononu traps enable|disable|auto slot/olt/onu phy|ont|line** command.

Note that only LineStatusTraps (i.e. line) has auto option.

```
zSH> gpononu traps disable 13/4/2 phy
```

Dynamic GEM ports

Delivering integrated and combined services such as IPTV, video on demand, data, voice over IP etc. requires enough network resource. To maintain control and be able to prioritize traffic in service-based order, Zhone OLT line card supports GPON Encapsulation Method (GEM) ports on ONUs.

GEM ports are dynamically created during the **bridge add** and **host add** operations. Conversely, GEM ports can be automatically deleted during the **bridge delete** and **host delete** operations. When creating a GEM port, a GPON traffic profile must be specified.

If an ONU is planning to be managed with Smart OMCI, when you creating a GEM port, make sure the GEM port ID range matches the GEM index that is selected in the Smart OMCI web-interface.

Each ONU supports up to 16 GPON GEM ports. The GEM port ID for each ONU combines the GEM index with the ONU ID:

User can select GEM index 5xx, 7xx, 9xx, etc. till 35xx in the Smart OMCI web-interface, where xx refers to the assigned ONU ID, which is in the range of 1 to 64. The 5, 7, 9...35 refers to the GEM index used by the ONU.

For example, GEM ports with GEM port IDs 501, 701, 901 are created on the ONU 1:

```
zSH> gpononu gemparts 4/1/1
```

ONU	GEM Port	Admin	traf	compn	share	Fixed UBR Bandwidth	Fixed CBR Bandwidth	Assured Bandwidth	Max Bandwidth	Extra Bandwidth	Type
allocId	DBA		prof			Mbits/sec	Mbits/sec	Mbits/sec	Mbits/sec	Mbits/sec	
1-4-1-11-4-1-901Up		3	False	False	1.024	0	0	1.024	Nonassured		
256	SR										
1-4-1-501Up		5	False	False	0.512	0	2.560	3.072	Nonassured		
501	SR										
1-4-1-701Up		4	False	False	0.256	0	0	0.256	Nonassured		
257	SR										

This section includes the following topics:

- [Create a GEM port with a GPON traffic profile, page451](#)
- [View the GEM port-related information, page458](#)

Create a GEM port with a GPON traffic profile

This section describes how to create GEM port on the bridge and route for data, voice, and video services.

Creating bridges on GEM ports for data, voice, and video services

This procedure describes how to use the **bridge add** command to create bridges on GEM ports to pass traffic between the MXK and the downlink ONUs for triple-play service. For different services, you can associate different GPON traffic profile with the GEM port.



Note: When creating multiple VLANs on same GEM port, the GTP must be the same. Otherwise the command will be rejected.

This section also describes how to configure an uplink bridge to pass traffic between the MXK and the upstream data/voice/video source.

The traffic class and its bandwidth are configured in the GPON traffic profile. The following examples show that GEM port 501 is configured for data service, and associated with GPON traffic profile 1; GEM port 701 is configured for voice service, and associated with GPON traffic profile 2; GEM port 901 is configured for video service, and associated with GPON traffic profile 3.

This ONU is managed with Smart OMCI, so the GEM index 5xx, 7xx, and 9xx match the GEM index that is selected in the Smart OMCI web-interface.

For more information on how to configure video bridging, see [MXK bridged video on page331](#).

1 Create a bridging configuration for data services:

```
zSH> bridge add 1-a-2-0/eth uplink vlan 100                                uplink bridge
zSH> bridge-path add ethernet2-100/bridge vlan 100 default                uplink bridge path
zSH> bridge add 1-13-1-501/gponport gtp 1 downlink vlan 100 tagged        downlink
GEM port
```

2 Create a bridging configuration for voice services:

```
zSH> bridge add 1-a-2-0/eth uplink vlan 200                                uplink bridge
zSH> bridge-path add ethernet2-200/bridge vlan 200 default                uplink bridge path
zSH> bridge add 1-13-1-701/gponport gtp 2 downlink vlan 200 tagged        downlink
GEM port
```

3 Create a bridging configuration for video services:

```
zSH> bridge add 1-a-2-0/eth uplink vlan 300                                uplink bridge
zSH> bridge-path add ethernet2-300/bridge vlan 300 default igmp timer 30  uplink bridge path
igmp snooping enable
zSH> bridge add 1-13-1-901/gponport gtp 3 downlink vlan 300 tagged video 1/6
downlink GEM port
```

- 4 View the newly created GEM ports and associated traffic profiles for selected ONU with the **gpononu gemports** command.

```
zSH> gpononu gemports 13/1/1
```

ONU	GEM Port	Admin	traf	compn	share	Fixed UBR Bandwidth Mbits/sec	Fixed CBR Bandwidth Mbits/sec	Assured Bandwidth Mbits/sec	Max Bandwidth Mbits/sec	Extra Bandwidth
1-13-1-11-13-1-501	1Up	1	False	False	0.512	0	n/a	n/a	n/a	
501	n/a									
1-13-1-701	2Up	2	True	False	0	0.512	n/a	n/a	n/a	
701	n/a									
1-13-1-901	3Up	3	True	False	0	0.512	n/a	n/a	n/a	
901	n/a									

Creating host-based routes on GEM ports for data, voice, and video services

This procedure describes how to use the **host add** command to create host-based routes on GEM ports to pass traffic between the MXK and downlink ONUs. This section also describes how to create a route on the uplink card to pass traffic between the MXK and the upstream data/voice/video source.

For the detail description of how to configure video service on routers, refer to [Configure host-based routing for video with local DHCP on page 316](#).

The following examples create host-based routes on GEM ports, and configure the MXK as a local DHCP server.

1 Create routed data services

```
zSH> interface add 1-a-2-0/eth vlan 100 125.1.1.1/24 creates IP interface on uplink
```

```
zSH> route add default 125.1.1.2541 metric value 1 specifying prefer use default route
```

```
zSH> interface add float data 125.1.2.254 255.255.255.0 creates a floating IP interface for data communication between the MXK, data source, and DHCP server
```

```
zSH> new dhcp-server-subnet 1 runs DHCP locally on MXK
```

```
dhcp-server-subnet 1
```

```
Please provide the following: [q]uit.
```

```
network: -----> {0.0.0.0}: 125.1.2.0
netmask: -----> {0.0.0.0}: 255.255.255.0
domain: -----> {0}:
range1-start: -----> {0.0.0.0}: 125.1.2.1
range1-end: -----> {0.0.0.0}: 125.1.2.250
range2-start: -----> {0.0.0.0}:
range2-end: -----> {0.0.0.0}:
range3-start: -----> {0.0.0.0}:
range3-end: -----> {0.0.0.0}:
range4-start: -----> {0.0.0.0}:
range4-end: -----> {0.0.0.0}:
```

```

default-lease-time: ----> {-1}:
min-lease-time: -----> {-1}:
max-lease-time: -----> {-1}:
boot-server: -----> {0.0.0.0}:
bootfile: -----> {}:
default-router: -----> {0.0.0.0}: 125.1.2.254 References the floating IP interface
primary-name-server: ---> {0.0.0.0}: 125.3.1.254
secondary-name-server: -> {0.0.0.0}: 125.3.1.253
domain-name: -----> {}:
subnetgroup: -----> {0}: 1 DHCP subnet group name
stickyaddr: -----> {enable}:
external-server: -----> {0.0.0.0}:
external-server-alt: ---> {0.0.0.0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
New record saved.

```

zSH> host add 1-13-1-501/gponport gtp 1 vlan 500 dynamic 1 1 *Creates downlink GEM port to pass data traffic between MXK and ONU*

The above example creates GEM 501 on ONU 1/13/1/1 for data services, assigns the GEM 501 GPON traffic profile 1, and uses VLAN 500. The **dynamic 1** refers to the DHCP subnet group 1, the second 1 refers to the numbers of floating IP addresses allowed.

2 Create routed voice services:

zSH> interface add 1-a-2-0/eth vlan 200 10.2.1.1/24 *creates IP interface on uplink*

zSH> route add 10.5.1.0 255.255.255.0 10.2.1.254 1

zSH> interface add float voice 10.3.1.254 255.255.255.0

```

zSH> new dhcp-server-subnet 2
dhcp-server-subnet 1
Please provide the following: [q]uit.
network: -----> {0.0.0.0}: 10.3.1.0
netmask: -----> {0.0.0.0}: 255.255.255.0
domain: -----> {0}:
range1-start: -----> {0.0.0.0}: 10.3.1.1
range1-end: -----> {0.0.0.0}: 10.3.1.250
range2-start: -----> {0.0.0.0}:
range2-end: -----> {0.0.0.0}:
range3-start: -----> {0.0.0.0}:
range3-end: -----> {0.0.0.0}:
range4-start: -----> {0.0.0.0}:
range4-end: -----> {0.0.0.0}:
default-lease-time: ----> {-1}:
min-lease-time: -----> {-1}:
max-lease-time: -----> {-1}:
boot-server: -----> {0.0.0.0}:
bootfile: -----> {}:
default-router: -----> {0.0.0.0}: 10.3.1.254 References the floating IP interface
primary-name-server: ---> {0.0.0.0}: 125.3.1.254

```

```

secondary-name-server: -> {0.0.0.0}: 125.3.1.253
domain-name: -----> {}:
subnetgroup: -----> {0}: 2 DHCP subnet group name
stickyaddr: -----> {enable}:
external-server: -----> {0.0.0.0}:
external-server-alt: ---> {0.0.0.0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
New record saved.

```

zSH> host add 1-13-1-701/gponport gtp 2 vlan 700 dynamic 2 1 *Creates downlink GEM port to pass voice traffic between MXK and ONU*

The above example creates GEM 701 on ONU 1/13/1/1 for voice services, assigns GEM 701 GPON traffic profile 2, and uses VLAN 700. The **dynamic 2** refers to the DHCP subnet group 2. **1** refers to the numbers of floating IP addresses allowed.

3 Create routed video services

zSH> interface add 1-a-2-0/eth vlan 300 10.4.1.1/24

zSH> video-source add 1-a-2-0/eth vlan 300 10.4.1.254 *creates a video source connection between the IP interface and the multi-cast address*

zSH> interface add float video 10.5.1.254 255.255.255.0

```

zSH> new dhcp-server-subnet 3
dhcp-server-subnet 1
Please provide the following: [q]uit.
network: -----> {0.0.0.0}: 10.5.1.0
netmask: -----> {0.0.0.0}: 255.255.255.0
domain: -----> {}:
range1-start: -----> {0.0.0.0}: 10.5.1.1
range1-end: -----> {0.0.0.0}: 10.5.1.250
range2-start: -----> {0.0.0.0}:
range2-end: -----> {0.0.0.0}:
range3-start: -----> {0.0.0.0}:
range3-end: -----> {0.0.0.0}:
range4-start: -----> {0.0.0.0}:
range4-end: -----> {0.0.0.0}:
default-lease-time: ----> {-1}:
min-lease-time: -----> {-1}:
max-lease-time: -----> {-1}:
boot-server: -----> {0.0.0.0}:
bootfile: -----> {}:
default-router: -----> {0.0.0.0}: 10.5.1.254 References the floating IP interface
primary-name-server: ---> {0.0.0.0}: 125.3.1.254
secondary-name-server: -> {0.0.0.0}: 125.3.1.253
domain-name: -----> {}:
subnetgroup: -----> {0}: 3 DHCP subnet group name
stickyaddr: -----> {enable}:
external-server: -----> {0.0.0.0}:
external-server-alt: ---> {0.0.0.0}:

```

```
.....
Save changes? [s]ave, [c]hange or [q]uit: s
New record saved.
```

zSH> host add 1-13-1-901/gponport gtp 3 vlan 900 dynamic 3 4 video 1/6 *Creates downlink GEM port to pass video traffic between MXK and ONU*

The above example creates GEM 901 for video services, assigns GEM 901 GPON traffic profile 3, and uses VLAN 900. The **dynamic 3** refers to the DHCP subnet group 3, **4** refers to the numbers of floating IP addresses allowed. The **1** in the **video 1/6** is the multicast control list index, and **6** is the maximum number of IP video streams.

- 4 View the newly created GEM ports and associated traffic profiles for the selected ONU with the **gpononu gemports** command.

zSH> gpononu gemports 13/1/1

ONU	GEM Port	traf	Admin	prof	compn	share	Fixed Bandwidth Mbits/sec	UBR Fixed Bandwidth Mbits/sec	CBR Assured Bandwidth Mbits/sec	Max Bandwidth Mbits/sec	Extra Bandwidth Mbits/sec	Type
1-13-1-11-13-1-501	501	Up	1	False	False	0.512	0	n/a	n/a	n/a		
1-13-1-701	701	Up	2	True	False	0	0.512	n/a	n/a	n/a		
1-13-1-901	901	Up	3	True	False	0	0.512	n/a	n/a	n/a		

Connection Admission Control (CAC) validation during the bridge/host creation on a GEM port

When using the **bridge add** or **host add** command to add a GEM port, or add a bridge/host to an existing GEM port with different VLAN, the following Connection Admission Control (CAC) validations are performed:

- Total available GEM ports should not exceed the maximum GEM ports.
Each ONU can support maximum of 16 GEM ports. If the value is greater then the error message "CAC Validation Error: The maximum allowed GEM ports of <value> exceeded" appears.
- Total available Alloc-Ids should not exceed the allowed values. The maximum # of Alloc-ids for Dynamic Bandwidth Allocation (DBA) is 384. The total # of Alloc-Id allowed is 768 (includes non-DBA and DBA).

If the validation fails, the error message "CAC Validation Error: The maximum allowed Alloc-ids <value> exceeded" appears. If the user tried to configure DBA, then 384 is displayed as the <value>. If configure non-DBA, then 768 is displayed as the <value>.

- The guaranteed, assured, and fixed class of services on the GPON physical port (i.e. OLT interface) should not exceed allowable bandwidth. The allowable bandwidth is the PON overhead subtracted from 1.248Gig per GPON physical port for upstream bandwidth. The average PON overhead is 110M.
- The total bandwidth on all the GEM ports on the GPON physical port should not exceed the total available bandwidth.
If the validation fails, the error message "CAC Validation Error: The total available bandwidth was exceeded" appears.
- Each GPON physical port can support 454246 Kbps available bandwidth for CBR. This bandwidth will be reduced by the UBR allocations that exceed 681,370 kbps.
- There is a 5% overhead for all DBA bandwidth allocations.
- Enabling OLT US FEC Parity will decrease available bandwidth by 145Mb/sec.

Before using the **bridge add** and **host add** command to create a GEM port, user can use the following two commands to check the available Alloc-Ids and available bandwidth on the GPON physical port.

Check the available Alloc-Ids on a GPON physical port with the **gponolt status gtp** command:

zSH> **gponolt status gtp**

OLT Interface	OLT State	# GEM Ports	DBA Alloc-Ids		Total Alloc-Ids	
			used	avail	used	avail
5/1	Active	0	0	384	0	768
6/1	Active	1 0		384	1	767
6/2	Ready	1	1	383	1	767
6/3	Inactive	2	1	383	2	766
6/4	Active	4	0	384	1	767

It also shows the OLT state. The possible values of the OLT state are:

Active: SFP is connected, fiber is connected, and active ONU is connected.

Ready: SFP is connected but no light seen on fiber.

Inactive: No SFP connected.

Check the available bandwidth on a GPON physical port with the **gponolt show bw** command:

zSH> **gponolt show bw 7/7**

SLOT 7/OLT 7:

Total Available BW..... 1090560 Kbps

```

Total Available BW for Compensated CBR... 454246 Kbps
Allocated UBR BW..... 32768 Kbps
Allocated CBR BW..... 0 Kbps
Allocated Compensated CBR BW..... 0 Kbps
Allocated Assured BW..... 0 Kbps
Allocated Non-Assured BW..... 0 Kbps
Allocated Best-Effort BW..... 0 Kbps

```

View the GEM port-related information

View the GEM port related information with the **gpononu gempports** command.

zSH> **gpononu gempports 7/3/1**

```

                                Fixed UBR Fixed CBR Assured   Max      Extra
                                Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU   GEM Port Admin prof compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type
allocId DBA
=====
=====
1-7-3-1 1-7-3-501   Up  1024 False False      1.024      0      n/a      n/a      n/a
501     n/a
      1-7-3-901   Up   512 False False      0.512      0      n/a      n/a      n/a
640     n/a

```

Table 23: Gpononu gempports Command Output Field Explanations

Field	Description
Onu	The ONU interface name in the format of shelf ID-Slot ID-OLT ID-ONU ID.
GEM Port	The ONU GEM port name in the format of shelfID-SlotID-OLT ID-ONU GEM Port ID.
Admin	The administrative status of the ONU
traf prof	The traffic profile index applied to the GEM port.
Compn	The compensation mode specified in the GPON traffic profile of the GEM port. Values: True False
Share	The shared mode specified in the GPON traffic profile of the GEM port. Values: True False
Fixed UBR Bandwidth Mbits/ sec	Fixed UBR bandwidth used when DBA is enabled.
Fixed CBR Bandwidth Mbits/ sec	Fixed CBR bandwidth used when DBA is enabled.

Table 23: Gpononu gempports Command Output Field Explanations

Field	Description
Assured Bandwidth Mbits/sec	DBA Assured bandwidth will be allocated when traffic demand exists.
Max Bandwidth Mbit/sec	Use this parameter to indicate the amount of non-guaranteed bandwidth configured for the traffic profile. Only available when DBA is enabled.
Extra Bandwidth Type	The priority type of non-guaranteed bandwidth. Only available when DBA is enabled.
allocID	The Alloc-Id assigned on this GEM port. If DBA is enabled, then this Alloc-Id is DBA enabled Alloc-Id, otherwise it is non-DBA Alloc-Id.
DBA	The DBA type chosen by system. Values: SR indicates the DBA type is Status Reporting and there is no error NSR indicates the DBA type is Non- Status Reporting and there is no error. Note: In the current release, only NSR is supported. NSR-Error indicates the DBA type NSR and there is an error in either getting the report from the ONU or the ONU does not support NSR. n/a indicates DBA has been disabled on the GEM port, or cannot communicate with ONU, or ONU has not been added. Error Indicates there are some errors

Bandwidth allocation

The bandwidth allocation for upstream traffic from ONU to MXK is configured in the GPON Traffic Profile (GTP).

This section includes the following topics:

- [Configure GPON traffic profile, page459](#)
- [Dynamic Bandwidth Allocation \(DBA\), page468](#)

Configure GPON traffic profile

The upstream bandwidth values, class of service types, and bandwidth allocation type are specified in a GTP.

Each GEM port has one GPON traffic profile.

One GPON traffic profile can be shared by multiple GEM ports.

For the detail DBA configuration on the GPON traffic profile, refer to section [Dynamic Bandwidth Allocation \(DBA\) on page468](#).

The CAC validation is performed during the GPON traffic profile configuration.

Creating a GPON traffic profile

Table24 provides description for GPON GEM ports upstream parameters.

Table 24: GPON GEM port uplink parameters description

Parameter	Description
guaranteed-upstream-bw	Specifies the upstream bandwidth for the GEM port. The value should be multiple of 512 and cannot exceed 1,048,576 Kbps. Note that the guaranteed-upstream-bw is for CBR and UBR only and not used in DBA. If DBA is disabled the guaranteed upstream bandwidth should be non-zero.
traffic-class	Specifies the upstream traffic class value. This is not used in DBA. Default: ubr Values: cbr Constant Bit Rate. The CBR class of traffic is used by connections that require a constant and guaranteed rate. The sampling time for CBR is constant, with no delay. CBR would be appropriate for real-time application, such as voice and video applications. ubr Unspecified Bit Rate. The UBR class of traffic does not specify traffic-related guarantees. No numerical commitments are made with respect to the packet loss or delay. With UBR service, the available bandwidth is fairly distributed to the active UBR subscribers. UBR would be appropriate for delay-tolerant or non-real-time applications, such as data application.
compensated	CBR compensation mode. Sometimes CBR access will be skipped after OLT and ONU exchanged GPON OAM messages. If you select true in compensated mode, CBR access can be compensated immediately after exchange of the GPON OAM messages to prevent possible jitter of the CBR channel. Compensation mode is used for both DBA and non-DBA. Default: false Values: true false

Table 24: GPON GEM port uplink parameters description

Parameter	Description
shared	Shared feature is to let the GEM ports to share upstream bandwidth. Select true if the GEM port which uses this traffic descriptor shares an Alloc-Id with another GEM port under the same ONU. False otherwise. Shared mode is used for both DBA and non-DBA. Default: false Values: true false
dba-enable	Enable or disable DBA. The default value is false. Default: false Values: true false
dba-fixed-us-ubr-bw	Fixed UBR bandwidth used when DBA is enabled. The minimum values of Fixed Ubr bandwidth can be 0 or 128 Kbps. The maximum value is 1,048,576 Kbps. Only multiples of 64 Kbps are allowed. Default: 0
dba-fixed-us-cbr-bw	Fixed CBR bandwidth used when DBA is enabled. The minimum value of Fixed Cbr bandwidth can be 0 or 512 Kbps. The maximum value is 454,208 Kbps. Only multiples of 64 Kbps are allowed. Default: 0
dba-assured-us-bw	DBA Assured bandwidth will be allocated when traffic demand exists. The minimum value of Assured bandwidth can be 0 or 256 Kbps. The maximum value is 1,048,576 Kbps. Only multiples of 64 Kbps are allowed. Default: 0

Table 24: GPON GEM port uplink parameters description

Parameter	Description
dba-max-us-bw	Use this parameter to indicate the amount of non-guaranteed bandwidth configured for the traffic profile. The non-guaranteed class of service can be either nonassured or besteffort type of service, which is specified in the dba-extra-us-bw-type field. The value of the non-guaranteed bandwidth can be computed using the configured value in this parameter minus the sum of the dba-fixed-us-ubr-bw, dba-fixed-us-cbr-bw and dba-assured-us-bw. The configured value in this parameter has to be greater than or equal to sum of the dba-fixed-us-ubr-bw, dba-fixed-us-cbr-bw and dba-assured-us-bw. If the configured value is equal to the sum of the dba-fixed-us-ubr-bw, dba-fixed-us-cbr-bw and dba-assured-us-bw, then no bandwidth is assigned to non-guaranteed type of service. The maximum value is 1,048,576 Kbps. Only multiples of 64 Kbps are allowed. Default: 0
dba-extra-us-bw-type	The priority type of non-guaranteed bandwidth. Default: nonassured Values: nonassured Nonassured has higher priority for getting unused bandwidth than besteffort. besteffort Besteffort has the lowest priority.

- 1 Create a GPON traffic profile with the **new gpon-traffic-profile** *index* command.

System provides the profile validation to ensure that the specified bandwidth value does not exceed the maximum value:

- If DBA is disabled, profile validation is performed to ensure that the static guaranteed-upstream-bandwidth does not exceed the maximum value.
- If DBA is enabled, profile validation is performed to ensure the each DBA related bandwidth does not exceed the maximum values, and the sum of dba-fixed-us-ubr-bw, dba-fixed-us-cbr-bw and dba-assured-us-bw does not exceed the dba-max-us-bw for the GTP profile.

```
mxk7-zSH> new gpon-traffic-profile 512

gpon-traffic-profile 512
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}: 2600000
```

```

Invalid entry: guaranteed-upstream-bw range: [0 to
1048576]
                                     profile validation on the value range
guaranteed-upstream-bw: -> {512}: 512
traffic-class: -----> {ubr}: cbr
compensated: -----> {false}: true
shared: -----> {false}:
dba-enabled: -----> {false}:
dba-fixed-us-ubr-bw: ----> {0}:
dba-fixed-us-cbr-bw: ----> {0}:
dba-assured-us-bw: -----> {0}:
dba-max-us-bw: -----> {0}:
dba-extra-us-bw-type: ---> {nonassured}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

```

- 2 View the GEM port parameter settings in a GPON traffic profile with the **get gpon-traffic-profile** *index* command.

```

mxk7-zSH> get gpon-traffic-profile 512
gpon-traffic-profile 512
guaranteed-upstream-bw: -> {512}
traffic-class: -----> {cbr}
compensated: -----> {true}
shared: -----> {false}
dba-enabled: -----> {false}:
dba-fixed-us-ubr-bw: ----> {0}:
dba-fixed-us-cbr-bw: ----> {0}:
dba-assured-us-bw: -----> {0}:
dba-max-us-bw: -----> {0}:
dba-extra-us-bw-type: ---> {nonassured}:

```

Sharing the Alloc-Id among GEM ports

The share feature in the GTP profile is to let the GEM ports to share upstream bandwidth.

The system supports up to 384 DBA Alloc-Ids per GPON physical port, and 768 Alloc-Ids per GPON physical port (including static and DBA).

Multiple GEM ports can share a single Alloc-Id if

- those GEM ports use the same traffic profile,
 - the traffic profile has "shared" set to true,
 - and those GEM ports are under the same ONU.

To turn on the share function, set the *shared* parameter to **true** in the GPON traffic profile.

- 1 View the Alloc-Id values assigned to the GEM ports when shared feature is disabled.

This example shows GEM ports 1-6-2-501 and 1-6-2-701 have different Alloc-Ids, 501 and 701.

zSH> **gpononu gempports6/2**

Processing list of 64

This command may take several minutes to complete.

Do you want to continue? (yes or no) [**yes**]

ONU	GEM Port	Admin	prof	compn	share	Fixed UBR Bandwidth	Fixed CBR Bandwidth	Assured Bandwidth	Max Bandwidth	Extra Bandwidth	Type
1-6-2-1	1-6-2-501	Up	1024	False	False	1.024	0	n/a	n/a	n/a	
501	n/a										
701	1-6-2-701	Up	1024	False	False	1.024	0	n/a	n/a	n/a	
1-6-2-2	1-6-2-502	Up	1024	False	False	1.024	0	n/a	n/a	n/a	
502	n/a										
702	1-6-2-702	Up	1024	False	False	1.024	0	n/a	n/a	n/a	
702	n/a										

Total Available BW: 1130.663(Mb), Total Available BW for Compensated CBR: 454.246 (Mb)

2 Create a GTP and enable the shared feature:

zSH> **new gpon-traffic-profile 512**

```
gpon-traffic-profile 512
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}: 512
traffic-class: -----> {cbr}: ubr
compensated: -----> {true}: false
shared: -----> {false}: true
dba-enabled: -----> {false}:
dba-fixed-us-ubr-bw: ----> {0}:
dba-fixed-us-cbr-bw: ----> {0}:
dba-assured-us-bw: -----> {0}:
dba-max-us-bw: -----> {0}:
dba-extra-us-bw-type: ---> {nonassured}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

3 Apply the GTP to multiple GEM ports by using the **bridge add** or **host add** command.

4 List all the ONU GEM ports use this GTP.

zSH> **gpononu gtp list 512**

To Abort the operation enter Ctrl-C

GEM Ports that use Traffic Profile 512

ONU Interface	GEM Port


```
=====
1-6-1-1 1-6-1-501
1-6-1-1 1-6-1-701
1-6-1-2 1-6-1-502
1-6-1-2 1-6-1-702
```

- 5 View the Alloc-Id values assigned to the GEM ports when shared feature is enabled.

This example shows GEM ports 1-6-1-501 and 1-6-1-701 have the same Alloc-Ids, 501.

```
zSH> gpononu gempports6/1
```

Processing list of 64

This command may take several minutes to complete.

Do you want to continue? (yes or no) [~~yes~~]

ONU	GEM Port	Admin	traf prof	Bandwidth	traf	Mbits/sec	class	compn	share	allocId	DBA
1-6-1-1	1-6-1-501	Up	512	0.5	ubrFalseTrue					501	n/a
	1-6-1-701	Up	512	0.5	ubrFalseTrue					501	n/a
1-6-1-2	1-6-1-502	Up	512	0.5	ubrFalseTrue					502	n/a
	1-6-1-702	Up	512	0.5	ubrFalseTrue					502	n/a

Total bandwidth: allocated = 2(Mb) used = 2(Mb)

Modifying a GPON traffic profile

Modify a GPON traffic profile with the **update gpon-traffic-profile** *index* command. You can only modify a GPON traffic profile that is not being used by a GEM port.

Modify a GPON traffic profile:

```
mxk7-zSH> update gpon-traffic-profile 512
```

```
gpon-traffic-profile 512
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {512}:
traffic-class: -----> {cbr}: ubr
compensated: -----> {true}: false
shared: -----> {false}:
dba-enabled: -----> {false}:
dba-fixed-us-ubr-bw: ----> {0}:
dba-fixed-us-cbr-bw: ----> {0}:
dba-assured-us-bw: -----> {0}:
dba-max-us-bw: -----> {0}:
dba-extra-us-bw-type: ---> {nonassured}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

The profile validation checks to see if the profile is being used by ONU GEM port. A GTP profile is considered as “in-use” if it is already assigned to a GEM port. If a GTP profile is in-use, the GTP modification is rejected, and an error message appears.

```
mxk7-zSH> update gpon-traffic-profile 513
gpon-traffic-profile 513
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {512}:
traffic-class: -----> {cbr}: ubr
compensated: -----> {true}: false
shared: -----> {false}:
dba-enabled: -----> {false}:
dba-fixed-us-ubr-bw: ----> {0}:
dba-fixed-us-cbr-bw: ----> {0}:
dba-assured-us-bw: -----> {0}:
dba-max-us-bw: -----> {0}:
dba-extra-us-bw-type: ---> {nonassured}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Profile Validation Error. The GTP profile is in use
and cannot be modified.

Starting over....
```

Deleting a GPON traffic profile

Delete a GPON traffic profile with the **delete gpon-traffic-profile index** command. You can only delete a GPON traffic profile that is not being used by a GEM port.

Delete a GPON traffic profile:

```
mxk7-zSH> delete gpon-traffic-profile 512
gpon-traffic-profile 512
1 entry found.

Delete gpon-traffic-profile 512? [y]es,[n]o,[q]uit: y
gpon-traffic-profile 512 deleted.
```

If a GTP profile is in-use, the deletion is rejected by the profile validation, and an error message appears.

```
zSH> delete gpon-traffic-profile 514
gpon-traffic-profile 514
1 entry found.
Delete gpon-traffic-profile 514? [y]es, [n]o, [q]uit :
yes
Profile Validation Error. The GTP profile is in use
and cannot be deleted.
gpon-traffic-profile 514 not deleted.
```

Viewing the existing GPON traffic profiles

View the existing GPON traffic profiles in the system with the **list gpon-traffic-profile** command.

```
mxk7-zSH> list gpon-traffic-profile
gpon-traffic-profile 1
gpon-traffic-profile 1024
gpon-traffic-profile 2
```

Viewing the GEM ports that use the same GPON traffic profile

View the GEM ports that use the same GPON traffic profile with the **gpononu gtp list GTPId** command.

```
zSH> gpononu gtp list 512
To Abort the operation enter Ctrl-C
GEM Ports that use Traffic Profile 512
ONU Interface      GEM Port
=====
1-13-1-1 1-13-1-501
1-7-7-3          1-7-7-503
1-7-7-4          1-7-7-504
1-7-7-5          1-7-7-505
```

Modifying the GPON traffic profile index of a GEM port

The GTP index assigned to a GEM port can be modified by the **update gpon-port-config** command. The new GTP index is specified in the **traffic-profile** parameter.

When modifying a GTP index of a GEM port, the profile validation compares the new GTP with the GTP assigned on the GEM port. The GTP index modification is rejected when any of the following conditions are met:

- If the DBA is disabled on both GTP profiles, and Traffic Class is changed.

The error message "Profile Validation Error: Cannot change the GTP index as this causes a change of Class of Service" appears.

- If the status of DBA is changed.

The error message "Profile Validation Error: Cannot change the GTP index as this causes change of DBA status." appears.

If the profile validation is successful, then the CAC validation is performed. The total available bandwidth is recalculated on the GPON physical port using the newly assigned GTP index. If the CAC validation fails, the error message "CAC Validation Error: The total available bandwidth exceeded" appears.

```

zSH> update gpon-port-config 1-13-1-501/gponport
Please provide the following: [q]uit.
multicast: -> {false}:
encrypted: -> {false}:
direction: -> {bidirectional}:
traffic-profile: -> {1}: 512
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Dynamic Bandwidth Allocation (DBA)

Dynamic Bandwidth Allocation (DBA) is specified in the ITU standard 984.3. This feature is used to grant upstream bandwidth to ONUs based on their demand and service-level agreement. The OLT will grant ONUs an increase in their slot time for more bandwidth while granting a reduced slot time to others. Through DBA, a GPON link can be oversubscribed for upstream traffic, and improve bandwidth usage efficiency.

Configuring DBA in the GPON traffic profile

User can enable or disable DBA and configure the DBA bandwidth in the GPON traffic profile.

User can configure both guaranteed (i.e. dba-fixed-us-ubr-bw, dba-fixed-us-cbr-bw, dba-assured-us-bw) and non-guaranteed DBA bandwidth (i.e. dba-max-us-bw). The non-guaranteed DBA bandwidth is always the UBR type of traffic and is non-compensated.

For the description of the DBA related GPON GEM ports uplink parameters, refer to [Table 24 on page 460](#).

Create a GPON-traffic-profile with DBA-related parameters:

```

zSH> new gpon-traffic-profile 430080
gpon-traffic-profile 430080
Please provide the following: [q]uit.
guaranteed-upstream-bw: -> {0}: For CBR and UBR only, not
used in DBA.
traffic-class: -----> {ubr}: For CBR and UBR only, not
used in DBA
compensated: -----> {false}: For DBA and non-DBA
shared: -----> {false}: For DBA and non-DBA
dba-enabled: -----> {false}: True DBA only
dba-fixed-us-ubr-bw: ----> {0}: 512 DBA only
dba-fixed-us-cbr-bw: ----> {0}: DBA only
dba-assured-us-bw: -----> {0}: 1024 DBA only
dba-max-us-bw: -----> {0}: 1024000 DBA only
dba-extra-us-bw-type: ---> {nonassured}: DBA only
.....
Save changes? [s]ave, [c]hange or [q]uit: s

```

Verify the settings:

```

zSH> get gpon-traffic-profile 430080
gpon-traffic-profile 430080
guaranteed-upstream-bw: -> {0}:
traffic-class: -----> {ubr}:
compensated: -----> {false}:
shared: -----> {false}:
dba-enabled: -----> {true}:
dba-fixed-us-ubr-bw: ----> {512}:
dba-fixed-us-cbr-bw: ----> {0}:
dba-assured-us-bw: -----> {1024}:
dba-max-us-bw: -----> {1024000}:
dba-extra-us-bw-type: ---> {nonassured}:

```

Configuring DBA in the GPON-OLT-Config profile

You can also find DBA parameters in the GPON-OLT-Config profile.

[Table 25](#) provides description for DBA related parameters. You can specify the values for those parameters in a gpon-olt-config profile.

Table 25: DBA related parameters description

Parameter	Description
dba-cycle	The Dynamic Bandwidth Allocation (DBA) cycle determines the DBA interval size. Its an integer multiplicand of 2 mili-seconds. The range of values for the DBA cycle can be between 2ms-10ms.
sr-dba-reporting-block-size	For DBA of SR type, the block size defines the resolution of the status reports per PON link. The default value is 48 bytes.

Edit the DBA parameter in a GPON OLT config profile:

```

zSH> update gpon-olt-config 1-11-1-0/gponolt
gpon-olt-config 1-11-1-0/gponolt
Please provide the following: [q]uit.
max-rt-propagation-delay: ----> {200}:
max-onu-response-time: -----> {50}:
preassigned-eqd: -----> {0}:
los-alpha: -----> {4}:
lof-alpha: -----> {4}:
loam-alpha: -----> {3}:
scrambler: -----> {enabled}:
fec-mode: -----> {disabled}:
auto-learn: -----> {enabled}:
power-level: -----> {0}:
guard-bit-count: -----> {32}:
dba-mode: -----> {predictive}:
gem-block-size: -----> {16}:
us-ber-interval: -----> {5000}:
ds-ber-interval: -----> {5000}:
ber-sf-threshold: -----> {3}:
ber-sd-threshold: -----> {5}:

```

```

fec-request: -----> {disabled}:
key-exchange: -----> {disabled}:
min-rt-propagation-delay: ----> {0}:
min-onu-response-time: -----> {10}:
eqd-measure-cycles: -----> {5}:
drift-ctrl-interval: -----> {1000}:
drift-ctrl-limit: -----> {3}:
alloc-cycle-length: -----> {2}:
min-us-alloc: -----> {16}:
ack-timeout: -----> {2000}:
pls-max-alloc-size: -----> {120}:
dba-cycle: -----> {2}: 8
sr-dba-reporting-block-size : -> {48}: 96
.....
Save changes? [s]ave, [c]hange or [q]uit: s

```

Verify the settings:

```

zSH> get gpon-olt-config 1-11-1-0/gponolt

gpon-olt-config 1-11-1-0/gponolt
max-rt-propagation-delay: ----> {200}:
max-onu-response-time: -----> {50}:
preassigned-eqd: -----> {0}:
los-alpha: -----> {4}:
lof-alpha: -----> {4}:
loam-alpha: -----> {3}:
scrambler: -----> {enabled}:
fec-mode: -----> {disabled}:
auto-learn: -----> {enabled}:
power-level: -----> {0}:
guard-bit-count: -----> {32}:
dba-mode: -----> {predictive}:
gem-block-size: -----> {16}:
us-ber-interval: -----> {5000}:
ds-ber-interval: -----> {5000}:
ber-sf-threshold: -----> {3}:
ber-sd-threshold: -----> {5}:
fec-request: -----> {disabled}:
key-exchange: -----> {disabled}:
min-rt-propagation-delay: ----> {0}:
min-onu-response-time: -----> {10}:
eqd-measure-cycles: -----> {5}:
drift-ctrl-interval: -----> {1000}:
drift-ctrl-limit: -----> {3}:
alloc-cycle-length: -----> {2}:
min-us-alloc: -----> {16}:
ack-timeout: -----> {2000}:
pls-max-alloc-size: -----> {120}:
dba-cycle: -----> {8}:
sr-dba-reporting-block-size : -> {96}:

```

Viewing the DBA type

There are two types of DBA, Status Reporting (SR) and Non-Status Reporting (NSR):

- SR: The ONU provides the bandwidth status information as part of the upstream traffic message. SR is specified in the Dynamic Bandwidth Report upstream (DBRu).
- NSR: NSR is the non-status reporting option where the OLT calculates the available bandwidth. The allocation is based on monitoring ONU's bandwidth usage compared to the allocated bandwidth.

DBA type is not selected by users. The system automatically chooses SR or NSR type. The system first attempts an SR connection, if the SR connection is not established, then the NSR connection is used. If SR is used to determine the DBA type, and subsequently the ONU sends error DBRu, after three consecutive DBRu received, the system switch to NSR. Because ONU shall be inactive when DBA type is changed, the only way to force back to SR is to use the **gpononu clear** command to de-register the ONU, after change use the **gpononu set** command to register the ONU again.

View the DBA type for an ONU with the **gpononu gemparts** command.

zSH> **gpononu gemparts 4/1/1**

ONU	GEM Port	Admin	prof	compn	share	Fixed UBR Bandwidth Mbits/sec	Fixed CBR Bandwidth Mbits/sec	Assured Bandwidth Mbits/sec	Max Bandwidth Mbits/sec	Extra Bandwidth	Type
allocId	DBA										
1-4-1-11-4-1-901Up		3	False	False	1.024	0	0	1.024	Nonassured		
256	SR										
1-4-1-501Up		5	False	False	0.512	0	2.560	3.072	Nonassured		
501	SR										
1-4-1-701Up		4	False	False	0.256	0	0	0.256	Nonassured		
257	SR										

OMCI Statistics

The MXK obtains ONU statistics from the ONU using OMCI. The MXK sends standards based OMCI commands to retrieve statistics information. The statistics are maintained on the ONU in 15-minute intervals. There are 2 intervals of statistics that is stored in the ONU, current and previous. When an ONU is activated, the ONU starts storing statistics. This statistics is stored under the current category of statistics. After a 15 minute time period, the statistics value are reset. The statistics tracked during the past 15 minute period are stored as the previous interval. A new set of the current interval statistics is tracked. After every 15-minute period the current interval is saved as previous and a new current category is created with zeroed out values.

Display OMCI statistics for selected ONU(s) with the **gpononu statistics** command.

Syntax:

gpononu statistics [**previous**] [*slot[/olt[/onu]]ifName*]

previous

System retrieves the statistics collected during the previous 15 minutes interval. Without **previous**, system retrieves the statistics collected in current 15 minutes interval.

slot[/olt[/onu]]ifName

The ONU(s) you want to collect statistics on.

Example:

```

zSH> gpononu statistics previous 13/4/2
13/4/2 ONU Statistics (previous)
  Ethernet Performance Monitoring History Data - Port 1
    139 Interval Time
      0 Threshold Data Pointer
      0 FCS Errors
      0 Excessive Collision Counter
      0 Late Collision Counter
      0 Frame Too Long
      0 Buffer Overflows on Receive
      0 Buffer Overflows on Transmit
      0 Single Collision Frame Counter
      0 Multiple Collisions Frame Counter
      0 SQE Counter
      0 Deferred Transmission Counter
      0 Internal MAC Transmit Error Counter
      0 Carrier Sense Error Counter
      0 Alignment Error Counter
      0 Internal MAC Receive Error Counter
  Ethernet Performance Monitoring History Data 2 - Port 1
    no data available
  GEM Port Protocol Monitoring History Data - Port 1
    139 Interval Time
      0 Threshold Data Pointer
      0 Lost packets
      0 Misinserted packets
      0 Rx Packets
      0 Rx Blocks
      0 Tx Blocks
      0 Impaired Blocks
  GEM Port Protocol Monitoring History Data - Port 2
    139 Interval Time
      0 Threshold Data Pointer
      0 Lost packets
      0 Misinserted packets
      0 Rx Packets
      0 Rx Blocks
      0 Tx Blocks
      0 Impaired Blocks
  GEM Port Protocol Monitoring History Data - Port 3

```



```

139 Interval Time
    0 Threshold Data Pointer
    0 Lost packets
    0 Misinserted packets
    0 Rx Packets
    0 Rx Blocks
    0 Tx Blocks
    0 Impaired Blocks
GEM Port Protocol Monitoring History Data - Port 4
139 Interval Time
    0 Threshold Data Pointer
    0 Lost packets
    0 Misinserted packets
    0 Rx Packets
    0 Rx Blocks
    0 Tx Blocks
    0 Impaired Blocks

```

[Table 26](#) defines the OMCI statistics displayed in the **gpononu statistics** command.

Table 26: OMCI statistics attributes

Attribute	Description
Interval end time	This attribute identifies the most recently finished 15-minute interval.
Threshold data 1/2 id	This attribute points to an instance of the threshold data 1 and 2 managed entities that contains Performance Monitoring threshold values.
FCS errors	This attribute counts frames received on a particular interface that were an integral number of octets in length but failed the frame check sequence (FCS) check. The count is incremented when the MAC service returns the frameCheckError status to the link layer control (LLC) or other MAC user. Received frames for which multiple error conditions are obtained are counted according to the error status presented to the LLC.
Excessive collision counter	This attribute counts frames whose transmission failed due to excessive collisions.
Late collision counter	This attribute counts the number of times that a collision was detected later than 512 bit times into the transmission of a packet.
Frames too long	This attribute counts received frames that exceeded the maximum permitted frame size. The count is incremented when the MAC service returns the frameTooLong status to the LLC.
Buffer overflows on receive	This attribute counts the number of times that the receive buffer overflowed.
Buffer overflows on transmit	This attribute counts the number of times that the transmit buffer overflowed.
Single collision frame counter	This attribute counts successfully transmitted frames whose transmission was delayed by exactly one collision.

Table 26: OMCI statistics attributes

Attribute	Description
Multiple collisions frame counter	This attribute counts successfully transmitted frames whose transmission was delayed by more than one collision.
SQE counter	This attribute counts the number of times that the SQE test error message was generated by the PLS sublayer.
Deferred transmission counter	This attribute counts frames whose first transmission attempt was delayed because the medium was busy. The count does not include frames involved in collisions.
Internal MAC transmit error counter	This attribute counts frames whose transmission failed due to an internal MAC sublayer transmit error.

PON Statistics

This section includes the following topics:

- [Viewing OLT stats, page475](#)
- [Viewing ONU stats, page483](#)

PON statistics are the OLT or ONU statistics collected and reported by MXK OLT.

The Downstream stats are the stats that were sent from MXK to ONU, and the upstream stats was sent from ONU to MXK.

MXK OLT can report these stats types for an OLT interface: GPON physical layer stats for OLT (i.e. gpon), Ethernet layer stats (i.e. rmon), and interface layer stats (i.e. intf). The GPON physical layer stats are only available on OLT interface.

MXK OLT can report these stats types for an ONU interface: ONU physical layer stats for ONU (i.e. onu) and interface layer stats (i.e. intf). The ONU physical layer stats are only available on ONU interface.

Collects and display OLT and ONU statistics with the **port statistics** command when specifying an OLT or ONU interface in the *inName/Type*.

Syntax:

```
port stats ifName/Type StatsType
```

ifName

interface name, in the format of *shelfID-SlotID-OLTID-ONUID*.

Type

interface type. e.g. gponolt, gpononu, eth, linegroup, etc.

To display stats for OLT or ONU interface, you must use interface type **gponolt** or **gpononu**.

StatsType

statistics type. The possible stats types are:

- **intf**
refers to mib2 interface statistics. intf is the default value, if there is no stats type specified, system shows intf stats. It is valid for all interface type.
- **rmon**
refers to ethernet remote monitoring statistics. It is valid for ethernet or gponolt interface.
- **gpon**
refers to GPON OLT physical layer statistics. It is valid for **gponolt** interface only.
- **onu**
refers to GPON ONU physical layer statistics as reported by MXK OLT. It is valid for **gpononu** interface only.
- **all**
refers to all statistics relevant to the interface type.

Viewing OLT stats

- 1 View the OLT interfaces. This step is optional.

```
zSH> list if-translate
```

```
Processing list of 896
```

```
if-translate 1-a-1-0/eth
if-translate ethernet1/linegroup
if-translate 1-a-5-0/ethproxy
if-translate 1-a-2-0/eth
if-translate ethernet2/linegroup
if-translate 1-a-3-0/eth
if-translate ethernet3/linegroup
if-translate 1-a-4-0/eth
if-translate 1-a-5-0/eth
if-translate ethernet5/linegroup
if-translate 1-a-6-0/ipobridge
if-translate ipobridge/linegroup
if-translate 1-a-1-0/rs232
if-translate 1-a-1-0/aal5proxy
if-translate 1-a-1-0-aal5proxy/linegroup
if-translate 1-a-2-0/dspproxy
if-translate 1-a-2-0-dspproxy/linegroup
if-translate 1-a-1-0-aal5proxy/atm
if-translate 1-a-1-0-aal5proxy/aal5
if-translate 1-a-1-0-aal5proxy/rfc1483
```

```

if-translate 1-a-2-0-dspproxy/atm
if-translate 1-a-2-0-dspproxy/aal5
if-translate 1-4-1-0/eth
if-translate 1-4-1-0-eth/linegroup
if-translate 1-4-2-0/eth
if-translate 1-4-2-0-eth/linegroup
if-translate 1-4-3-0/eth
if-translate 1-4-3-0-eth/linegroup
if-translate 1-4-4-0/eth
if-translate 1-4-4-0-eth/linegroup
if-translate 1-4-5-0/ethproxy
if-translate 1-4-5-0/eth
if-translate 1-4-5-0-eth/linegroup
if-translate 1-4-6-0/eth
if-translate 1-4-6-0-eth/linegroup
if-translate 1-4-7-0/eth
if-translate 1-4-7-0-eth/linegroup
if-translate 1-4-8-0/eth
...
if-translate 1-7-3-0/gponolt
...
if-translate 1-7-3-1/gpononu
...
896 entries found.

```

- When specifying **all** as the stats type, the interface, rmon, and OLT stats are displayed for this OLT interface.

```

zSH> port stats 1-7-3-0/gponolt all
***** intf *****
Interface Name                1-7-3-0
Operational Status            Up
Received Bytes                 60751467067
Received Packets               888061705
Received Multicast Packets     2154983
Received Broadcast Packets     76
Transmitted Bytes              1315135012976
Transmitted Unicast Packets    886804106
Transmitted Multicast Packets  3522742
Transmitted Broadcast Packets  15602009
Received Discards              2162928
Received Errors                0
Received Unknown Protocols     0
Transmitted Discards           0
Transmitted Errors             0
Speed Bits per Second          *** n/a ***
Speed Megabits per Second      2400
***** rmon *****
Total Dropped Events           0
Total Dropped Frames           0
Total Bytes                    1375886488877
Total Packets                  1796145737
Transmitted Packets            905928953
Received Packets               890216784

```

Transmitted Multicast Bytes	0
Received Multicast Bytes	0
Received Multicast Dropped Bytes	0
Transmitted Average Throughput	0
Received Average Throughput	0
Transmitted Bandwidth Occupancy	0
Received Bandwidth Occupancy	0
Total Broadcast Packets	15602149
Total Multicast Packets	5677777
CRC Align Errors	0
Undersize Packets	0
Oversize Packets	0
Transmitted Oversize Packets	0
Received Oversize Packets	0
Fragments	0
Jabbers	0
Collisions	0
Transmitted No Errors	905928953
Received No Errors	890216784
IPMC Bridged Packets	0
IPMC Routed Packets	0
Transmitted IPMC Dropped Packets	0
Received IPMC Dropped Packets	0
Total Packets 0 to 64 Bytes	2512147
Total Packets 65 to 127 Bytes	904526445
Total Packets 128 to 255 Bytes	21353234
Total Packets 256 to 511 Bytes	227817
Total Packets 512 to 1023 Bytes	5849962
Total Packets 1024 to 1518 Bytes	861676132
Total Packets 1519 to 2047 Bytes	0
Total Packets 2048 to 4095 Bytes	0
Total Packets 4095 to 9216 Bytes	0
Received Packets 0 to 64 Bytes	2
Received Packets 65 to 127 Bytes	888733225
Received Packets 128 to 255 Bytes	1256099
Received Packets 256 to 511 Bytes	227458
Received Packets 512 to 1023 Bytes	0
Received Packets 1024 to 1518 Bytes	0
Received Packets 1519 to 2047 Bytes	0
Received Packets 2048 to 4095 Bytes	0
Received Packets 4095 to 9216 Bytes	0
Transmitted Packets 0 to 64 Bytes	2512145
Transmitted Packets 65 to 127 Bytes	15793220
Transmitted Packets 128 to 255 Bytes	20097135
Transmitted Packets 256 to 511 Bytes	359
Transmitted Packets 512 to 1023 Bytes	5849962
Transmitted Packets 1024 to 1518 Bytes	861676132
Transmitted Packets 1519 to 2047 Bytes	0
Transmitted Packets 2048 to 4095 Bytes	0
Transmitted Packets 4095 to 9216 Bytes	0
***** olt *****	
Upstream Valid Gem Frames	890216798
Upstream Discarded Frames	0
Upstream Gem Frames	890210014

Upstream OmcI Frames	6784
Upstream Ploam Frames	1705525846
Upstream Idle Ploam Frames	1704842590
Downstream Valid Gem Frames	905929019
Downstream Discarded Frames	3115
Downstream Gem Frames	905923060
Downstream OmcI Frames	5959
Downstream Ploam Frames	73094
Downstream Idle Ploam Frames	0
Downstream Pon Valid Ethernet Packtes	905919875
Downstream Pon Cpu Packets	5956
Downstream Transmitted Bytes	1315058058309
Upstream Pon Valid Packets	890210000
Upstream Pon Valid Not Idle Ploams	683256
Upstream Pon Error Ploams	12
Upstream Pon Invalid Packets	0
Upstream Dropped Packets	0
Upstream Dropped Ploams Fifo Full	0
Downstream TM Valid Packets	905929019
Downstream TM Crc Packets	0
Downstream TM Dropped Cpu Packets	0
Downstream TM MAC Lookup Miss	0
Downstream TM Packets Forwarded From Hm To Pon	645624943
Downstream TM Packets Dropped Gem Pid Not Enabled	3115
Downstream TM Q0 Valid Packets	905923060
Downstream TM Q0 Dropped Packets	0
Downstream TM Q1 Valid Packets	0
Downstream TM Q1 Dropped Packets	0
Downstream TM Q2 Valid Packets	0
Downstream TM Q2 Dropped Packets	0
Downstream TM Q3 Valid Packets	0
Downstream TM Q3 Dropped Packets	0
Downstream TM Q4 Valid Packets	0
Downstream TM Q4 Dropped Packets	0
Downstream TM Q5 Valid Packets	0
Downstream TM Q5 Dropped Packets	0
Downstream TM Q6 Valid Packets	0
Downstream TM Q6 Dropped Packets	0
Downstream TM Q7 Valid Packets	0
Downstream TM Q7 Dropped Packets	0
Upstream TM Dropped Cpu Packets	0
Upstream TM Dropped Packets Crc Error	0
Upstream TM Dropped Packets Security	0
Upstream TM Learn Failures	0
Upstream TM Q0 Valid Packets	890210014
Upstream TM Q0 Dropped Packets	0
Upstream TM Q1 Valid Packets	0
Upstream TM Q1 Dropped Packets	0
Upstream TM Q2 Valid Packets	0
Upstream TM Q2 Dropped Packets	0
Upstream TM Q3 Valid Packets	0
Upstream TM Q3 Dropped Packets	0
Upstream TM Q4 Valid Packets	0
Upstream TM Q4 Dropped Packets	0

Upstream TM Q5 Valid Packets	0
Upstream TM Q5 Dropped Packets	0
Upstream TM Q6 Valid Packets	0
Upstream TM Q6 Dropped Packets	0
Upstream TM Q7 Valid Packets	0
Upstream TM Q7 Dropped Packets	0

- 3** When specifying **gpon** as the stats type, only the GPON OLT physical layer statistics are displayed for this OLT interface.

```
zSH> port stats 1-7-3-0/gponolt gpon
```

Upstream Valid Gem Frames	1390778452
Upstream Discarded Frames	0
Upstream Gem Frames	1390766390
Upstream Omci Frames	12062
Upstream Ploam Frames	2773259552
Upstream Idle Ploam Frames	2772149075
Downstream Valid Gem Frames	1408605291
Downstream Discarded Frames	3416
Downstream Gem Frames	1408595361
Downstream Omci Frames	9930
Downstream Ploam Frames	117890
Downstream Idle Ploam Frames	0
Downstream Pon Valid Ethernet Packtes	1408591816
Downstream Pon Cpu Packets	9930
Downstream Transmitted Bytes	2050960928301
Upstream Pon Valid Packets	1390766377
Upstream Pon Valid Not Idle Ploams	1110477
Upstream Pon Error Ploams	23
Upstream Pon Invalid Packets	0
Upstream Dropped Packets Inactive Ports	0
Upstream Dropped Ploams Fifo Full	0
Downstream TM Valid Packets	1408605259
Downstream TM Crc Packets	0
Downstream TM Dropped Cpu Packets	0
Downstream TM MAC Lookup Miss	0
Downstream TM Packets Forwarded From Hm To Pon	1005110436
Downstream TM Packets Dropped Gem Pid Not Enabled	3416
Downstream TM Q0 Valid Packets	1408595361
Downstream TM Q0 Dropped Packets	0
Downstream TM Q1 Valid Packets	0
Downstream TM Q1 Dropped Packets	0
Downstream TM Q2 Valid Packets	0
Downstream TM Q2 Dropped Packets	0
Downstream TM Q3 Valid Packets	0
Downstream TM Q3 Dropped Packets	0
Downstream TM Q4 Valid Packets	0
Downstream TM Q4 Dropped Packets	0
Downstream TM Q5 Valid Packets	0
Downstream TM Q5 Dropped Packets	0
Downstream TM Q6 Valid Packets	0
Downstream TM Q6 Dropped Packets	0
Downstream TM Q7 Valid Packets	0
Downstream TM Q7 Dropped Packets	0

```

Upstream TM Dropped Cpu Packets          0
Upstream TM Dropped Packets Crc Error    0
Upstream TM Dropped Packets Security      0
Upstream TM Learn Failures                0
Upstream TM Q0 Valid Packets              1390766390
Upstream TM Q0 Dropped Packets            0
Upstream TM Q1 Valid Packets              0
Upstream TM Q1 Dropped Packets            0
Upstream TM Q2 Valid Packets              0
Upstream TM Q2 Dropped Packets            0
Upstream TM Q3 Valid Packets              0
Upstream TM Q3 Dropped Packets            0
Upstream TM Q4 Valid Packets              0
Upstream TM Q4 Dropped Packets            0
Upstream TM Q5 Valid Packets              0
Upstream TM Q5 Dropped Packets            0
Upstream TM Q6 Valid Packets              0
Upstream TM Q6 Dropped Packets            0
Upstream TM Q7 Valid Packets              0
Upstream TM Q7 Dropped Packets            0

```

[Table27](#) defines the GPON OLT physical layer statistics displayed in the **port stats ifName/gponolt** command.

Note that the Downstream stats are the stats that were sent from MXK to ONU, and the upstream stats was sent from ONU to MXK.

Table 27: GPON OLT physical layer statistics attributes

Attribute	Description
Upstream Valid Gem Frames	The number of valid GEM frames sent in upstream direction.
Upstream Discarded Frames	Total number of discarded GEM frames sent in upstream direction.
Upstream Gem Frames	The number of GEM frames sent in the upstream direction.
Upstream Omci Framesr	The number of OMCI frames sent in the upstream direction.
Upstream Ploam Frames	Total number of Physical Layer Operations, Administration and Maintenance (PLOAM) frames sent in the upstream direction. This includes: • Total number of PLOAM messages, including idle PLOAMs. • Total number of valid PLOAM messages (not including idle PLOAMs) • Total number of PLOAM messages dropped due to Cyclic Redundancy Check (CRC) errors.
Upstream Idle Ploam Frames	Total number of idle PLOAM frames sent in upstream direction.

Table 27: GPON OLT physical layer statistics attributes

Attribute	Description
Downstream Valid Gem Frames	Total number of valid GEM frames sent in downstream direction.
Downstream Discarded Frames	The number of downstream packets discarded due to CRC errors, MAC lookup miss, congestion, etc.
Downstream Gem Frames	Total number of GEM frames sent in downstream direction.
Downstream Omci Frames	Total number of OMCI frames sent in downstream direction.
Downstream Ploam Frames	Total number of PLOAM frames sent in downstream direction.
Downstream Idle Ploam Frames	Total number of idle PLOAM frames sent in downstream direction.
Downstream Pon Valid Ethernet Packets	Total number of valid Ethernet packets sent in downstream direction.
Downstream Pon Cpu Packets	The number of downstream packets generated by the CPU (MIPS).
Downstream Transmitted Bytes	Total number of bytes transmitted sent in downstream direction.
Upstream Pon Valid Packets	Total number of valid PON packets sent in upstream direction.
Upstream Pon Valid Not Idle Ploams	Total number of valid non-idle PLOAM messages sent in upstream direction.
Upstream Pon Error Ploams	Total number of PON error PLOAM messages sent in upstream direction.
Upstream Pon Invalid Packets	The number of upstream errored packets.
Upstream Dropped Packets Inactive Ports	Total number of upstream packets that were dropped because the GEM port ID was not configured.
Upstream Dropped Ploams Fifo Full	Total number of upstream PLOAMs that were dropped because the FIFO buffer was full.
Downstream TM Valid Packets	Total number of valid packets that were sent in downstream direction.
Downstream TM Crc Packets	The number of dropped downstream packets due to CRC errors.

Table 27: GPON OLT physical layer statistics attributes

Attribute	Description
Downstream TM Dropped Cpu Packets	The number of dropped downstream packets originated by the CPU (MIPS).
Downstream TM MAC Lookup Miss	The number of downstream MAC lookup miss events.
Downstream TM Packets Forwarded From Hm To Pon	The number of downstream packets forwarded from the header modification stage to the PON.
Downstream TM Packets Dropped Gem Pid Not Enabled	The number of downstream packets dropped because the GEM port ID was not configured correctly.
Downstream TM QN Valid Packets (N=0 to 7)	The number of downstream packets forwarded by egress priority queue [0 to 7] to the PON. Queue 0 is the highest priority; queue 7 is the lowest priority. Packets in the lowest priority queues are dropped first. When the PON link is not active, this counter will not increment.
Downstream TM QN Dropped Packets (N=0 to 7)	The number of downstream packets dropped by egress priority queue [0 to 7] due to congestion. Queue 0 is the highest priority; queue 7 is the lowest priority. Packets in the lowest priority queues are dropped first. When the PON link is not active, this counter will not increment.
Upstream TM Dropped Cpu Packets	The number of upstream packets dropped by the CPU(MIPS), not sent to SGMI interface.
Upstream TM Dropped Packets Crc Error	The number of upstream packets that were dropped because of CRC errors.
Upstream TM Dropped Packets Security	Total number of upstream packets that were dropped because they didn't pass the security rules.
Upstream TM Learn Failures	MAC address learning failures from traffic sent in upstream direction that were due to a full FIFO buffer.
Upstream TM QN Valid Packets (N=0 to 7)	Number of upstream packets forwarded by egress priority queue [0 to 7] to the MxK. Queue 0 is the highest priority; queue 7 is the lowest priority. Packets in the lowest priority queues are dropped first. When the PON link is not active, this counter will not increment.
Upstream TM QN Dropped Packets (N=0 to 7)	Number of upstream packets dropped by egress priority queue [0 to 7] due to congestion. Queue 0 is the highest priority; queue 7 is the lowest priority. Packets in the lowest priority queues are dropped first. When the PON link is not active, this counter will not increment.

Viewing ONU stats

- 1 View the ONU interfaces. This step is optional.

```
zSH> list if-translate
```

```
Processing list of 896
```

```
if-translate 1-a-1-0/eth
if-translate ethernet1/linegroup
if-translate 1-a-5-0/ethproxy
if-translate 1-a-2-0/eth
if-translate ethernet2/linegroup
if-translate 1-a-3-0/eth
if-translate ethernet3/linegroup
if-translate 1-a-4-0/eth
if-translate 1-a-5-0/eth
if-translate ethernet5/linegroup
if-translate 1-a-6-0/ipobridge
if-translate ipobridge/linegroup
if-translate 1-a-1-0/rs232
if-translate 1-a-1-0/aal5proxy
if-translate 1-a-1-0-aal5proxy/linegroup
if-translate 1-a-2-0/dspproxy
if-translate 1-a-2-0-dspproxy/linegroup
if-translate 1-a-1-0-aal5proxy/atm
if-translate 1-a-1-0-aal5proxy/aal5
if-translate 1-a-1-0-aal5proxy/rfc1483
if-translate 1-a-2-0-dspproxy/atm
if-translate 1-a-2-0-dspproxy/aal5
if-translate 1-4-1-0/eth
if-translate 1-4-1-0-eth/linegroup
if-translate 1-4-2-0/eth
if-translate 1-4-2-0-eth/linegroup
if-translate 1-4-3-0/eth
if-translate 1-4-3-0-eth/linegroup
if-translate 1-4-4-0/eth
if-translate 1-4-4-0-eth/linegroup
if-translate 1-4-5-0/ethproxy
if-translate 1-4-5-0/eth
if-translate 1-4-5-0-eth/linegroup
if-translate 1-4-6-0/eth
if-translate 1-4-6-0-eth/linegroup
if-translate 1-4-7-0/eth
if-translate 1-4-7-0-eth/linegroup
if-translate 1-4-8-0/eth
...
if-translate 1-7-3-0/gponolt
...
if-translate 1-7-3-1/gpononu
...
896 entries found.
```

- 2 When specifying **onu** as the stats type, the ONU physical layer statistics are displayed for this ONU interface.

```

zSH> port stats 1-7-3-3/gpononu onu
Upstream BIP8 Errors                                0
Upstream FEC Corrected Bytes                        0
Upstream FEC Corrected Code-words                  0
Upstream FEC Uncorrectable Code-words               0
Upstream Received Code-words                       0
Upstream Unreceived Bursts                          0
Downstream Remote BIP8 Errors                       0
Upstream Remote BIP8 Errors                         0
Drift Of Window Indications                         0
Message Error Message                               0

```

- 3 When there are no stats type specified or only specifying **intf**, the interface statistics are displayed for this ONU interface.

```

zSH> port stats 1-7-3-3/gpononu
Interface Name                                     1-7-3-3
Operational Status                               Up
Received Bytes                                    0
Received Packets                                  0
Received Multicast Packets                        0
Received Broadcast Packets                        0
Transmitted Bytes                                 0
Transmitted Unicast Packets                       0
Transmitted Multicast Packets                     0
Transmitted Broadcast Packets                     0
Received Discards                                 0
Received Errors                                   0
Received Unknown Protocols                        0
Transmitted Discards                              0
Transmitted Errors                                0
Speed Bits per Second                             0
Speed Megabits per Second                         0

```

- 4 When specifying **all** as the stats type, both interface and ONU stats are displayed for this ONU interface.

```

zSH> port stats 1-7-3-3/gpononu all
***** intf *****

Interface Name                                     1-7-3-3
Operational Status                               Up
Received Bytes                                    0
Received Packets                                  0
Received Multicast Packets                        0
Received Broadcast Packets                        0
Transmitted Bytes                                 0
Transmitted Unicast Packets                       0
Transmitted Multicast Packets                     0
Transmitted Broadcast Packets                     0

```

```

Received Discards                                0
Received Errors                                  0
Received Unknown Protocols                       0
Transmitted Discards                             0
Transmitted Errors                               0
Speed Bits per Second                           0
Speed Megabits per Second                       0

```

```
*****  onu  *****
```

```

Upstream BIP8 Errors                            0
Upstream FEC Corrected Bytes                    0
Upstream FEC Corrected Code-words               0
Upstream FEC Uncorrectable Code-words           0
Upstream Received Code-words                    0
Upstream Unreceived Bursts                      0
Downstream Remote BIP8 Errors                   0
Upstream Remote BIP8 Errors                     0
Drift Of Window Indications                     0
Message Error Message                           0

```

[Table28](#) defines the GPON ONU physical layer statistics displayed in the **port stats ifName/gpononu** command.

Table 28: GPON ONU physical layer statistics attributes

Attribute	Description
Upstream BIP8 Errors	Total number of upstream Bit-Interleaved Parity with eight bit (BIP8) errors per ONU-ID.
Upstream FEC Corrected Bytes	Total number of upstream FEC corrected bytes per ONU-ID.
Upstream FEC Corrected Code-words	Total number of upstream FEC corrected code words per ONU-ID.
Upstream FEC Uncorrectable Code-words	Total number of upstream FEC uncorrected code words per ONU-ID.
Upstream Received Code-words	Total number of upstream code words transmitted per ONU-ID.
Upstream Unreceived Bursts	Total number of upstream un-received bursts per ONU-ID.
Downstream Remote BIP8 Errors	Total number of downstream remote BIP8 errors per ONU-ID.
Upstream Remote BIP8 Errors	Total number of upstream remote BIP8 errors per ONU-ID.

Table 28: GPON ONU physical layer statistics attributes

Attribute	Description
Drift Of Window Indications	The number of times the average drift for the ONU exceeds the drift threshold.
Message Error Message	The number of error messages send from ONU.

10 MXK 20-PORT ACTIVE ETHERNET CARD

This chapter describes the MXK 20-port Active Ethernet card and Active Ethernet card configuration:

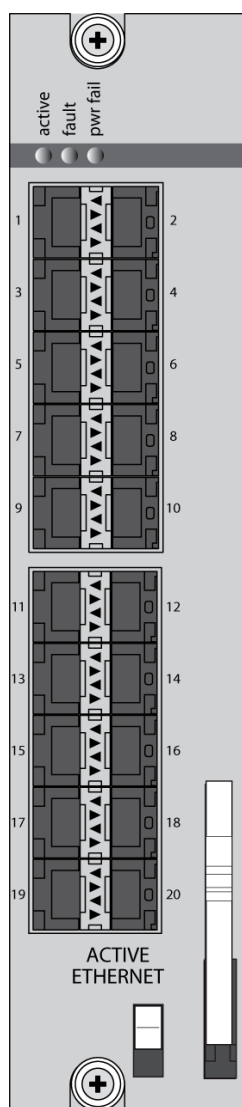
- [20-port Active Ethernet card, page488](#)
- [Small form factor pluggables, page492](#)
- [Displaying and updating Ethernet interfaces, page492](#)

20-port Active Ethernet card

This section describes the MXK 20-port dual-slot Active Ethernet card and Active Ethernet card configuration:

- [Active Ethernet card overview, page488](#)
- [Active Ethernet card specifications, page489](#)
- [Active Ethernet 20 port card configuration, page489](#)
- [View additional card and system information, page491](#)

Active Ethernet card overview



The MXK Active Ethernet line card, MXK-AEX20-FE/GE-2S, is a two slot card that supports Ethernet traffic over 20 ports that provide either 10/100/

1000 Base-T, fiber 100FX or 1 Gigabit Ethernet interfaces to support distances as high as 80km depending on the SFPs used. The Active Ethernet card is also interoperable third party Active Ethernet devices.

The Active Ethernet card supports Layer 2 bridging functions, Layer 2 security functions, Layer 3 routing functions and the Zhone Multimedia Traffic Management functionality (MTM).

This card supports non-redundant GigE connections to subtended SLMS devices.

Active Ethernet card specifications

[Table29](#) provides the Active Ethernet MXK-AEX20-FE/GE-2S card specifications.

Table 29: Active Ethernet MXK-AEX20-FE/GE-2S card specifications

Specification	Description
Size	2 slot
Density	20 GigE ports
Physical interfaces	10/100/1000 Ethernet ports with SFPs. The optical interfaces are class 1 Laser International Safety Standard IEC 825 compliant 20 Gigabit Ethernet ports with SFPs. The SFPs can be twisted pair 1000baseT or fiber (SX, LX or ZX). See Small form factor pluggables on page492 .
Standards supported	IEEE 802.3 IEEE 802.1 Q/P IEEE 802.1 AD (Q in Q)
Power consumption	52.3 W

Active Ethernet 20 port card configuration

Each card installed in the system must have a **card-profile**. The line card type determines the parameter settings in the **card-profile** and the software image for the card. Performing a **card add <slot #>** automatically creates the **card-profile** for the card with the correct software image and settings.

[Table30](#) provides the type and software image for the Active Ethernet cards on the MXK.

Table 30: MXK card type for Active Ethernet

Card	Type	Name of software image
MXK-AEX20-FE/GE-2S	10200	<i>mxlc20ae.bin</i>

Configuring an Active Ethernet card

To configure an Active Ethernet card on the system:

- 1 Install the Active Ethernet card in the desired line card slot.
- 2 Verify the card by entering **slots**:

```
zSH> slots
Uplinks
a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
4: MXK 4 PORT GPON (RUNNING)
13: MXK 20 ACT ETH (RUNNING)
```

- 3 Create a **card-profile** for the card:

```
zSH> card add 13
```

- 4 Verify the **card-profile** for the Active Ethernet card:

```
zSH> get card-profile 1/13/10200
card-profile 1/13/10200
sw-file-name: -----> {mxlc20ae.bin}
admin-status: -----> {operational}
upgrade-sw-file-name: ---> {}
upgrade-vers: -----> {}
admin-status-enable: ----> {enable}
sw-upgrade-admin: -----> {reloadcurrrev}
sw-enable: -----> {true}
sw-upgrade-enable: -----> {false}
card-group-id: -----> {0}
hold-active: -----> {false}
weight: -----> {nopreference}
card-line-type: -----> {unknown-type}
card-atm-configuration: -> {notapplicable}
card-line-voltage: -----> {not-used}
maxvpi-maxvci: -----> {notapplicable}
card-init-string: -----> {}
wetting-current: -----> {disabled}
```

- 5 Connect the line-side cables to the SFP connectors on the Active Ethernet card.

Verifying the line card installation

After you save the **card-profile** record, the line card in that slot resets and begins downloading the software image from the flash card. This could take a few moments.

When the card has finished loading, a log message similar to the following is displayed (if logging is enabled):

```
zSH> Card in slot slot-number changed state to RUNNING
```

- 1 To view the status of all the cards enter **slots**:

```
zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  4: MXK 4 PORT GPON (RUNNING)
 13: MXK 20 ACT ETH (NOT_PROV)
```

- 2 To view card information including the state of the card and how long the card has been running you enter **slots** and specify the slot number of the card:

```
zSH> slots 13
Type           : MXK 20 ACT ETH
Card Version    : 800-02316-01-A
EEPROM Version  : 1
Serial #        : 1769100
CLEI Code       : No CLEI
Card-Profile ID : 1/13/10200
Shelf           : 1
Slot            : 13
ROM Version     : MXK 1.16.0.128
Software Version: MXK 1.16.1.128
State           : RUNNING
Mode            : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat   : 80
Fault reset     : enabled
Uptime          : 19 days, 3 hours, 26 minutes
```

View additional card and system information

View the EPROM version of the card:

```
zSH> eeshow card 13
EEPROM contents: for slot 13
EEPROM_ID      : 00 -- CARD
Version        : 01
Size           : 054
CardType       : 10200 -- MXLC20AE
CardVersion    : 800-02316-01-A
SerialNum      : 01769100
ShelfNumber    : 00001
CLEI Code      : No CLEI
Cksum          : 0x4EB4
```

View the ROM version of the card:

```
zSH> romversion 13
MXK 1.16.0.128
Feb  9 2009, 20:10:55
```

Verify the existence of a daughter card on the system:

```
zSH> eeshow 1d
EEPROM contents: for slot 30
EEPROM_ID   : 01 -- 1DAUGHTER
Version     : 01
Size        : 022
CardType    : 10100 -- MXUP2TG8G
CardVersion : 800-02482-01-B
SerialNum   : 01768971
ShelfNumber : 00001
CLEI Code   : No CLEI
Cksum       : 0xFA1F
```

View the version of the system software:

```
zSH> swversion
Zhone mxUp2Tg8g software version MXK 1.16.1.128
```

Small form factor pluggables

Zhone Technologies supports a variety of small form factor pluggables (SFPs) and XFPs which are selected depending on the protocol, fiber type and distance requirements. For information and specifications on supported SFPs and XFPs, see [Chapter 15, Small Form Factor Pluggable \(SFP\) Connectors, on page757](#).

Displaying and updating Ethernet interfaces

The **list**, **get**, and **update** commands support use of the interface *shelf-slot-port-subport/eth* syntax to facilitate Ethernet port and interface monitoring and configuration.

To list the currently configured Ethernet interfaces, enter the **list ether** command.

```
zSH> list ether
ether 1-a-1-0/eth
ether 1-a-2-0/eth
ether 1-a-3-0/eth
ether 1-a-4-0/eth
ether 1-a-5-0/eth
ether 1-a-6-0/eth
ether 1-a-7-0/eth
ether 1-a-8-0/eth
ether 1-a-9-0/eth
ether 1-a-10-0/eth
ether 1-a-11-0/eth
ether 1-b-1-0/eth
ether 1-b-2-0/eth
ether 1-b-3-0/eth
ether 1-b-4-0/eth
```

```

ether 1-b-5-0/eth
ether 1-b-6-0/eth
ether 1-b-7-0/eth
ether 1-b-8-0/eth
ether 1-b-9-0/eth
ether 1-b-10-0/eth
ether 1-b-11-0/eth
ether 1-13-1-0/eth
ether 1-13-2-0/eth
ether 1-13-3-0/eth
ether 1-13-4-0/eth
...
42 entries found.

```

The **list ether** command shows the Ethernet interfaces on each uplink card as well as the Ethernet interfaces on the Active Ethernet card in slot 13.

The **slots** command verifies the location of the cards with Ethernet interfaces:

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  4: MXK 4 PORT GPON (RUNNING)
 13: MXK 20 ACT ETH (RUNNING)

```

To view an Ethernet interface, enter the **get ether** command followed by the interface/type.



Note: Link aggregation is not supported on Active Ethernet ports.

```

zSH> get ether 1-13-20-0/eth
ether 1-13-20-0/eth
autonegstatus: ----> {enabled}
mauType: -----> {mau1000basetfd}
restart: -----> {norestart}
ifType: -----> {mau1000basetfd}
autonegcap: -----> {b100baseTX+b100baseTXFD+b1000baseT+b1000baseTFD}
remotefault: -----> {noerror}
clksrc: -----> {automatic}
pauseFlowControl: -> {disabled}
aggregationMode: --> {off}
linkStateMirror: --> {0/0/0/0/0}

```


11 MXK 20-PORT SINGLE SLOT ACTIVE ETHERNET CARD

This chapter describes the MXK 20-port Active Ethernet single slot card and Active Ethernet card configuration:

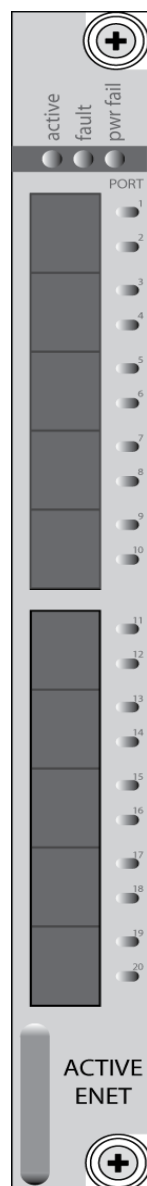
- [20-port single slot Active Ethernet card, page496](#)
- [Small form factor pluggables, page500](#)
- [Display and update Ethernet interfaces, page500](#)

20-port single slot Active Ethernet card

This section describes the single slot MXK 20-port Active Ethernet card and Active Ethernet card configuration:

- [Active Ethernet card overview, page496](#)
- [Active Ethernet card specifications, page497](#)
- [Active Ethernet 20-port single slot card configuration, page497](#)
- [View additional card and system information, page499](#)

Active Ethernet card overview



The MXK Active Ethernet line card, MXK-AEX20-FE/GE, is a single slot card that supports Ethernet traffic over 10 ports that provide either 10/100/1000 Base-T, fiber 100FX or 1 Gigabit Ethernet interfaces to support distances as high as 80km depending on the SFPs used. The Active Ethernet card is also interoperable third party Active Ethernet devices.

The Active Ethernet card supports Layer 2 bridging functions, Layer 2 security functions, Layer 3 routing functions and the Zhone Multimedia Traffic Management functionality (MTM).

This card supports non-redundant GigE connections to subtended SLMS devices.

Active Ethernet card specifications

[Table31](#) provides the Active Ethernet card specifications.

Table 31: Active Ethernet card specifications

Specification	Description
Size	1 slot
Density	20 GigE ports
Physical interfaces	10/100/1000 Ethernet ports with SFPs. The optical interfaces are class 1 Laser International Safety Standard IEC 825 compliant 20 Gigabit Ethernet ports with SFPs. The SFPs can be twisted pair 1000baseT or fiber (SX, LX or ZX). See Chapter 15, Small Form Factor Pluggable (SFP) Connectors , on page757.
Standards supported	IEEE 802.3 IEEE 802.1 Q/P IEEE 802.1 AD (Q in Q)
Power consumption	52.3 W

Active Ethernet 20-port single slot card configuration

Each card installed in the system must have a **card-profile**. The line card type determines the parameter settings in the **card-profile** and the software image for the card. Performing a **card add <slot #>** automatically creates the **card-profile** for the card with the correct software image and settings.

[Table32](#) shows the type and software image for the Active Ethernet card on the MXK.

Table 32: MXK card type for Active Ethernet

Card	Type	Name of software image
MXK-AEX20-FE/GE	10207	mxlc20ae1s.bin

Adding Active Ethernet cards

To add an Active Ethernet card to the system:

- 1 Install the Active Ethernet card in the desired line card slot.
- 2 Create a **card-profile** for the card:

```
zSH> card add 12
```

After performing a **card add** in a slot, the slot resets and begins downloading the software image from the flash card. This could take a few moments.

When the card has finished loading, a log message similar to the following is displayed (if logging is enabled):

```
zSH> Card in slot slot-number changed state to RUNNING
```

- 3 Verify the card by entering **slots**:

```
zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  7: MXK 20 ACT ETH SINGLE SLOT (RUNNING)
  9: MXK 20 ACT ETH (RUNNING)
 11: MXK 8 PORT GPON (RUNNING)
 12: MXK 20 ACT ETH SINGLE SLOT (RUNNING)
 13: MXK 20 ACT ETH (RUNNING)
```

- 4 Verify the **card-profile** for the Active Ethernet card:

```
zSH> get card-profile 1/12/10207
card-profile 1/12/10207
sw-file-name: -----> {mxlc20ae1s.bin}
admin-status: -----> {operational}
upgrade-sw-file-name: ---> {}
upgrade-vers: -----> {}
admin-status-enable: ----> {enable}
sw-upgrade-admin: -----> {reloadcurrrev}
sw-enable: -----> {true}
sw-upgrade-enable: -----> {false}
card-group-id: -----> {0}
hold-active: -----> {false}
weight: -----> {nopreference}
card-line-type: -----> {unknowntype}
card-atm-configuration: -> {notapplicable}
card-line-voltage: -----> {not-used}
maxvpi-maxvci: -----> {notapplicable}
card-init-string: -----> {}
wetting-current: -----> {disabled}
```

- 5 View card information including the state of the card and how long the card has been running:

```

zSH> slots 12
Type           : MXK 20 ACT ETH SINGLE SLOT
Card Version   : 800-02703-01-Z
EEPROM Version : 1
Serial #       : 9999006
CLEI Code      : PROT006A08
Card-Profile ID : 1/12/10207
Shelf          : 1
Slot           : 12
ROM Version    : development
Software Version: MXK 1.16.1.211
State          : RUNNING
Mode           : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat  : 348
Fault reset    : enabled
Uptime         : 5 days, 3 hours, 18 minutes

```

- 6 Connect the line-side cables to the SFP connectors on the Active Ethernet card.

View additional card and system information

View the EPROM version of the card:

```

zSH> eeshow card 12
EEPROM contents: for slot 12
EEPROM_ID      : 00 -- CARD
Version        : 01
Size           : 054
CardType       : 10207 -- MXLC20AE_SINGLE_SLOT
CardVersion    : 800-02703-01-Z
SerialNum      : 09999006
ShelfNumber    : 00001
CLEI Code      : PROT006A08
Cksum          : 0x7B81

```

View the ROM version of the card:

```

zSH> romversion 12
development
Feb  9 2009, 09:01:01

```

Verify the existence of a daughter card on the system:

```

zSH> eeshow 1d
EEPROM contents: for slot 30
EEPROM_ID      : 01 -- 1DAUGHTER
Version        : 01
Size           : 022
CardType       : 10100 -- MXUP2TG8G
CardVersion    : 800-02482-01-D
SerialNum      : 01762719
ShelfNumber    : 00001

```

CLEI Code : No CLEI
Cksum : 0x1289

Verify the existence of a second daughter card by entering **eeshow 2d**.

View the version of the system software:

```
zSH> swversion
Zhone mxUp2Tg8g software version MXK 1.16.2.127
```

Small form factor pluggables

Zhone Technologies supports a variety of small form factor pluggables (SFPs) and XFPs which are selected depending on the protocol, fiber type and distance requirements.

The MXK-AEX20-FE/GE single slot Active Ethernet card supports bi-directional SFPs. For information and specifications on these supported SFPs, see [Chapter 15, Small Form Factor Pluggable \(SFP\) Connectors, on page 757](#).

Display and update Ethernet interfaces

The **list**, **get**, and **update** commands support use of the interface *shelf-slot-port-subport/eth* syntax to facilitate Ethernet port and interface monitoring and configuration.

To list the currently configured Ethernet interfaces, enter the **list ether** command.

```
zSH> list ether
ether 1-a-1-0/eth
ether 1-a-2-0/eth
ether 1-a-3-0/eth
ether 1-a-4-0/eth
ether 1-a-5-0/eth
ether 1-a-6-0/eth
ether 1-a-7-0/eth
ether 1-a-8-0/eth
ether 1-a-9-0/eth
ether 1-a-10-0/eth
ether 1-a-11-0/eth
ether 1-b-1-0/eth
ether 1-b-2-0/eth
ether 1-b-3-0/eth
ether 1-b-4-0/eth
ether 1-b-5-0/eth
ether 1-b-6-0/eth
ether 1-b-7-0/eth
ether 1-b-8-0/eth
ether 1-b-9-0/eth
ether 1-b-10-0/eth
```

```
ether 1-b-11-0/eth
ether 1-12-1-0/eth
ether 1-12-2-0/eth
ether 1-12-3-0/eth
ether 1-12-4-0/eth
...
42 entries found.
```

The **list ether** command shows the Ethernet interfaces on each uplink card as well as the Ethernet interfaces on the Active Ethernet card in slot 12.

To view an Ethernet interface, enter the **get ether** command followed by the interface/type.



Note: Link aggregation is not supported on Active Ethernet ports.

```
zSH> get ether 1-12-1-0/eth
ether 1-12-1-0/eth
autonegstatus: ----> {enabled}
mauType: -----> {mau1000basetfd}
restart: -----> {norestart}
ifType: -----> {mau1000basetfd}
autonegcap: -----> {b100baseTX+b100baseTXFD+b1000baseT+b1000baseTFD}
remotefault: -----> {noerror}
clksrc: -----> {automatic}
pauseFlowControl: -> {disabled}
aggregationMode: --> {off}
linkStateMirror: --> {0/0/0/0/0}
```


12 MXK ADSL2+ BOND CARDS

This chapter describes the MXK ADSL2+ bond cards and ADSL card configuration:

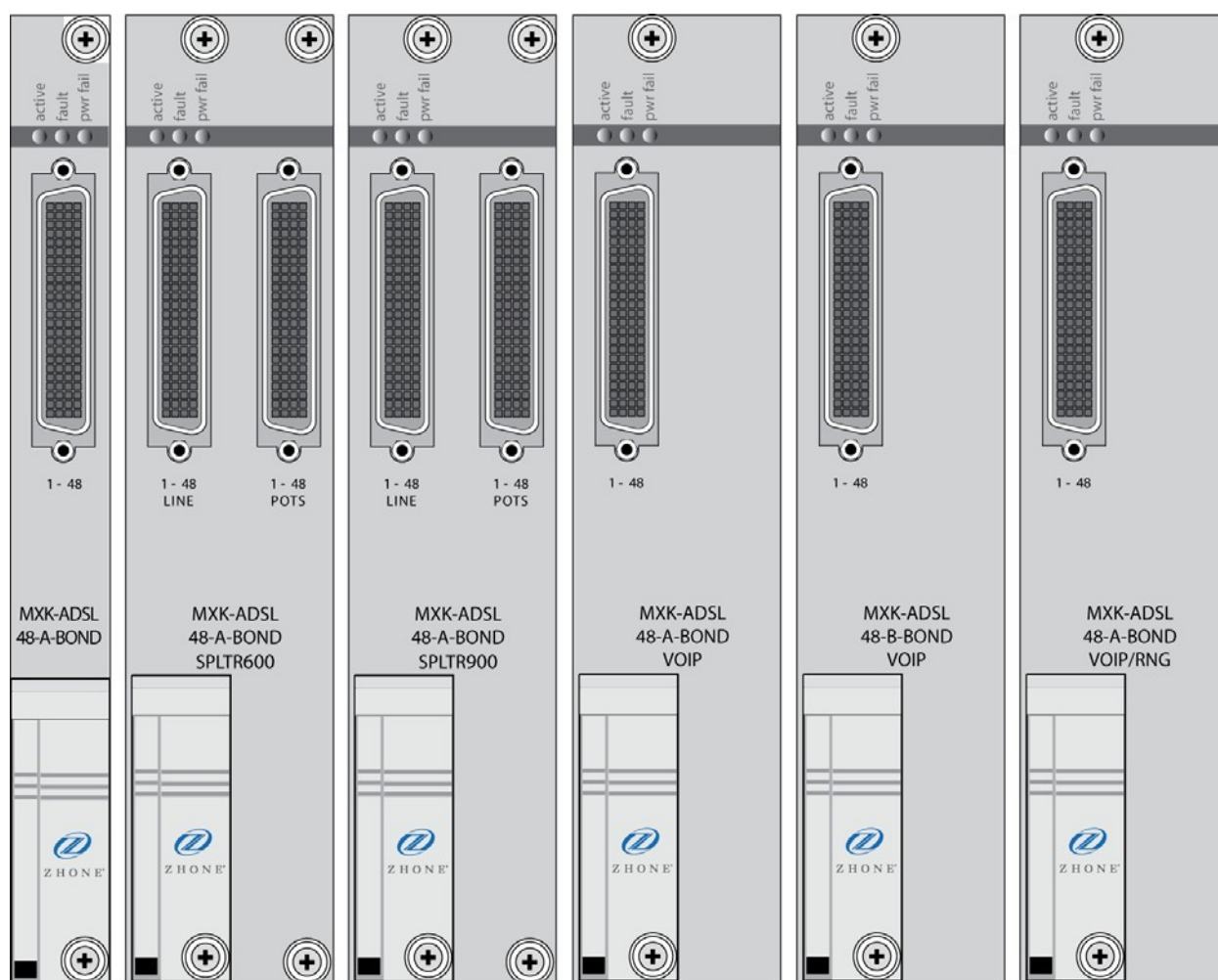
- [ADSL2+ bond cards, page504](#)
- [ADSL on the MXK, page510](#)
- [Configure ADSL2+ interfaces, page521](#)
- [ADSL2+ statistics, page555](#)
- [ADSL2+ bonding, page566](#)
- [ATM on the ADSL2+ POTS line card, page569](#)
- [POTS to Voice over IP \(VoIP\) configuration, page572](#)
- [Emergency Stand Alone \(ESA\) SIP support, page592](#)
- [ADSL2+ cable and port pinouts, page595](#)
- [ADSL2+ testing \(SELT/DELT\) on the MXK, page606](#)

ADSL2+ bond cards

This section describes the MXK ADSL2+ bond cards with splitters and without splitters and how to configure the ADSL interfaces.

- [ADSL2+ card overview, page504](#)
- [ADSL2+ card specifications, page506](#)
- [ADSL2+ bond card configuration, page508](#)
- [View additional card information, page510](#)

ADSL2+ card overview



The ADSL2+ bond cards are:

- MXK-ADSL2+-BCM-48A
- MXK-ADSL2+-POTS-BCM-48A-2S
- MXK-ADSL2+-POTS-BCM-48B-2S
- MXK-ADSL2+-POTS-BCM-48A-RNG-2S
- MXK-ADSL2+-SPLTR600-BCM-48A-2S
- MXK-ADSL2+-SPLTR900-BCM-48A-2S

MXK-ADSL2+-BCM-48A

This card is a single slot card that supports ADSL2+ Annex A/M. The standards supported are ANSI T1.413 Issue 2, G.992.1 (G.dmt), G.992.2 (G.lite), and ADSL2+ (G.992.5) standards.

MXK-ADSL2+-POTS-BCM-48A-2S

MXK-ADSL2+-POTS-BCM-48B-2S

These are two-slot cards and to provide 48 ports of integrated ADSL and POTS VoIP services. This card supports the ANSI T1.413 Issue 2, G.992.1 (G.dmt) and G.992.2 (G.lite), G.992.3 and G.992.4 (ADSL2), G.992.5 (ADSL2+), Annex A, Annex B, and Annex M ADSL standards. Also supports SIP, SIP-PLAR, H.248, and MGCP protocols.

MXK-ADSL2+-POTS-BCM-48A-RNG-2S

Integrated ringing functionality on the line card means that the total number of POTS lines on an MXK chassis that can be in the ringing state simultaneously is increased as well as simplifying the effort required to match the ringing capacity of the MTAC cards to the POTS lines on the shelf. Also, when the POTS lines in the chassis are provisioned on this card, a separate TAC/RING card is not needed in the chassis, which increases the line capacity of the shelf and reduces the per port costs of deployment.

MXK-ADSL2+-SPLTR600-BCM-48-2S

MXK-ADSL2+-SPLTR900-BCM-48-2S

These cards are two-slot cards with an integrated POTS splitter to provide 48 ports of integrated ADSL and POTS service. Each of these lines are combined with the ADSL2+ signal internally and exits the line card in the subscriber direction with both ADSL and POTS on the loop. In the network direction POTS is split from the ADSL signal keeping POTS on copper pairs and placing the ADSL data information on the IP network.

The MXK-ADSL2+-SPLTR600-BCM-48-2S, and MXK-ADSL2+-SPLTR900-BCM-48-2S cards support the ANSI T1.413 Issue 2, G.992.1 (G.dmt) and G.992.2 (G.lite), G.992.3 and G.992.4 (ADSL2), G.992.5 (ADSL2+), Annex A standards and Annex M ADSL standards.

The MXK-ADSL2+-BCM-48A, MXK-ADSL2+-SPLTR600-BCM-48-2S, and MXK-ADSL2+-SPLTR900-BCM-48-2S cards support VoIP POTS services.

Caution: Use caution when inserting and removing the ADSL2+ cards to and from the MXK chassis. Be careful not to damage the caps on the board.

ADSL2+ card specifications

[Table 33](#) provides the specifications for the MXK-ADSL2+-BCM-48A, MXK-ADSL2+-SPLTR600-BCM-48-2S, MXK-ADSL2+-SPLTR900-BCM-48-2S, MXK-ADSL2+-POTS-BCM-48A-2S, MXK-ADSL2+-POTS-BCM-48A-RNG-2S, and MXK-ADSL2+-POTS-BCM-48B-2S.

Table 33: ADSL-48-BCM cards specifications

Specification	Description
Size	1 slot: MXK-ADSL-BCM-48A 2 slot: MXK-ADSL2+-SPLTR600-BCM-48-2S, MXK-ADSL2+-SPLTR900-BCM-48-2S, MXK-ADSL2+-POTS-BCM-48A-2S, and MXK-ADSL2+-POTS-BCM-48B-2S MXK-ADSL2+-POTS-BCM-48A-RNG-2S
Density	48 ports ADSL
Connectors	1 slot: One 96-pin telco connector 2 slot: Two 96-pin telco connectors
Standards supported	ANSI T1.413.2 G.992.1 (G.DMT) G.992.2 (G.Lite) G.992.3 and G.992.4 (ADSL2) G.992.5 (ADSL2+)
Line characteristics	Annex A supported (ADSL2+ over POTS) Annex B supported (ADSL2+ over ISDN) Annex A/M supported (ADSL2/ADSL2+) Fast Path or Interleaved mode supported on a per port basis Fast Retrain supported

Table 33: ADSL-48-BCM cards specifications (Continued)

Specification	Description
Supported train rates	T1.413 (fast and interleaved): - up to 8192 Kbps downstream - up to 1024 Kbps upstream G.lite (interleaved only) - up to 1536Kbps downstream - up to 512Kbps upstream G.DMT (interleave): - up to 12288Kbps downstream - up to 1331.2Kbps upstream G.DMT (fast): - up to 80192Kbps downstream - up to 1024Kbps upstream ADSL2: - Annex A: up to 12288 Kbps downstream - Annex A: up to 1331.2 Kbps upstream - Annex B: 32 Kbps to 28 Mbps downstream 32 Kbps to 1.2 Mbps upstream ADSL2+: - Annex A: up to 28672Kbps downstream - Annex A: up to 1024Kbps upstream - Annex M: up to 24576Kbps downstream - Annex M: up to 3584Kbps upstream In the case of Annex M, the upstream will vary depending on how many bins are allocated from the downstream to the upstream. Typically the best case for increasing the upstream bandwidth is approximately 2252.8Kbps.
Metallic test function	Look-out test
Power ADSL2+	23 Watts nominal 38.16 W maximum total. This is at maximum distance with all ports trained at ADSL2+ rates
Power ADSL2+ splitter	23 Watts nominal 38.16 W maximum total. This is at maximum distance with all ports trained at ADSL2+ rates
Chip set	Broadcom

ADSL2+ bond card configuration

Each card installed in the system must have a **card-profile**. The line card type determines the parameter settings in the **card-profile** and the software image for the card. Performing a **card add <slot #>** automatically creates the **card-profile** for the card with the correct software image and settings.

[Table 34](#) shows the card type and software image for the ADSL2+ bond cards on the MXK.

Table 34: MXK ADSL2+ bond card types

Card	Type	Name of software image
MXK-ADSL2+-BCM-48A	10202	<i>mxlc48aadslbond.bin</i>
MXK-ADSL2+-SPLTR600-BCM-48-2S	10202	<i>mxlc48aadslbond.bin</i>
MXK-ADSL2+-SPLTR900-BCM-48-2S	10202	<i>mxlc48aadslbond.bin</i>
MXK-ADSL2+-POTS-BCM-48A-2S	10202	<i>mxlc48aadslbond.bin</i>
MXK-ADSL2+-POTS-BCM-48B-2S	10202	<i>mxlc48badslbond.bin</i>
MXK-ADSL2+-POTS-BCM-48A-RNG-2S	10202	

The types of 48-port ADSL2+ bonded cards on the MXK are:

- **unknowntype** (default)
- **adsl-pots-pv** (ADSL with VoIP)
- **adsl-splitter** (ADSL with splitters)

Adding ADSL2+ cards

To add an ADSL2+ card to the system:

- 1 Install the ADSL2+ card in the desired line card slot.
- 2 Create a **card-profile** for the card:

```
zSH> card add 1
```

After performing a **card add** in a slot, the slot resets and begins downloading the software image from the flash card. This could take a few moments.

When the card has finished loading, a log message similar to the following is displayed (if logging is enabled):

```
zSH> Card in slot slot-number changed state to RUNNING
```

- 3 Verify the card by entering **slots**:

```
zSH> slots
Uplinks
a:*MXK EIGHT GIGE (RUNNING)
```

```

b: MXK EIGHT GIGE (NOT_PROV)
Cards
1: MXK ADSL-48-A Bonded (RUNNING)
2: MXK ADSL-48-A Bonded (RUNNING)
3: MXK ADSL-48-A Bonded (RUNNING)
4: MXK ADSL-48-A Bonded (RUNNING)
5: MXK ADSL-48-A Bonded (RUNNING)
6: MXK ADSL-48-A Bonded (RUNNING)
7: MXK ADSL-48-A Bonded (RUNNING)
8: MXK ADSL-48-A Bonded (RUNNING)
9: MXK ADSL-48-A Bonded (RUNNING)
10: MXK ADSL-48-A Bonded (RUNNING)
11: MXK ADSL-48-A Bonded/with 900 Ohm Splitter (RUNNING)
13: MXK ADSL-48-A Bonded/with 900 Ohm Splitter (RUNNING)
15: MXK ADSL-48-A Bonded/with 600 Ohm Splitter (RUNNING)
17: MXK ADSL-48-A Bonded/with 600 Ohm Splitter (RUNNING)

```

4 Verify the **card-profile** for the ADSL card:

```

zSH> get card-profile 1/1/10202
card-profile 1/1/10202
sw-file-name: -----> {mxlc48aadslbond.bin}
admin-status: -----> {operational}
upgrade-sw-file-name: ---> {}
upgrade-vers: -----> {}
admin-status-enable: ----> {enable}
sw-upgrade-admin: -----> {reloadcurrrev}
sw-enable: -----> {true}
sw-upgrade-enable: -----> {false}
card-group-id: -----> {0}
hold-active: -----> {false}
weight: -----> {nopreference}
card-line-type: -----> {unknown-type}
card-atm-configuration: -> {notapplicable}
card-line-voltage: -----> {not-used}
maxvpi-maxvci: -----> {notapplicable}
card-init-string: -----> {}
wetting-current: -----> {disabled}
pwe-timing-mode: -----> {none}

```

5 View card information including the state of the card and how long the card has been running:

```

zSH> slots 1
Type           : MXK ADSL-48-A Bonded
Card Version    : 800-02775-01-B
EEPROM Version  : 1
Serial #       : 2368431
CLEI Code      : No CLEI
Card-Profile ID : 1/1/10202
Shelf          : 1
Slot           : 1
ROM Version    :
Software Version: release 1.16

```

```
State          : RUNNING
Mode           : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat  : 18616
Fault reset    : enabled
Uptime         : 18 hours, 45 minutes
```

View additional card information

View the EPROM version of the card:

```
zSH> eeshow card 1
EEPROM contents: for slot 1
EEPROM_ID      : 00 -- CARD
Version        : 01
Size           : 054
CardType       : 10202 -- MXLC48AADSLBOND
CardVersion    : 800-02775-01-B
SerialNum      : 02368431
ShelfNumber    : 00001
CLEI Code      : No CLEI
Cksum          : 0x1F8E
```

View the EPROM version of the daughter card:

```
zSH> eeshow 1d 9
EEPROM contents: for slot 9
EEPROM_ID      : 01 -- 1DAUGHTER
Version        : 01
Size           : 022
CardType       : 05080 -- MALC_ADSL_48_ANNEXA_M_BOND
CardVersion    : 800-02451-01-A
SerialNum      : 01578673
ShelfNumber    : 00001
CLEI Code      : No CLEI
Cksum          : 0xB00D
```

ADSL on the MXK

This section describes how ADSL functions on the MXK including:

- [ADSL overview, page511](#)
- [ADSL2+ transmission modes, page511](#)
- [ADSL2+ rate adaptation, page512](#)
- [Advanced ADSL2+ configurations on the MXK, page512](#)

ADSL overview

MXK ADSL interfaces provide a standards-based, high-speed DSL interface between the MXK and CPE devices.

Asymmetric Digital Subscriber Line (ADSL) is a type of DSL that uses existing telephone lines to solve the issue of first mile connection. It is more expensive to dig trenches and lay fiber than it is to deploy ADSL technology over twisted wire pairs of existing telephone lines.

This is possible because voice signals use the portion of the frequencies which can be sent over a twisted wire pair below 4kHz and ADSL uses the portion of the frequencies above 25kHz.

ADSL is asymmetric because the data flow is greater in one direction. The range of frequencies used by ADSL is separated into two frequency bands — the upstream band to the central office and the downstream band to the end user. The downstream band is larger, hence downloads to a home computer are faster than uploads.

Signals sent down copper wire may be impaired by distance from the central office, noise on the wire, and radio interference from AM radio stations.

ADSL devices can adjust to signal conditions to achieve the highest possible speeds, so usually no adjustment is needed. The ability to adjust to signal conditions is called “training.” The default settings used by ADSL cards on the MXK are suitable in most cases, although configuration options are available for fine tuning when necessary.

Configurable ADSL2+ options on the MXK pertain to the accuracy of transmission packets and overall throughput as shown in [Table35](#).

Table 35: Configurable ADSL2+ options

Option	Description
Signal to Noise Ratio	Provides a mechanism to adjust the robustness of the ADSL Link and hence the speed.
Transport Mode	Defines how packets are sent down the line. Fast provides a simple contiguous message which does not require much processing time to disassemble and reassemble packets. Interleaved provides greater protection from short bursts of noise that can result in lost packets
Bonding	Bonding is the ability to have multiple ports work together, so they appear as one larger pipe. ADSL bonding allows combining two ports.

ADSL2+ transmission modes

[Table36](#) describes the transmission modes MXK ADSL2+ cards support.

Table 36: Supported transmission modes

Transmission Mode	Description
ADSL2	The modem negotiates rates up to 12 Mbps downstream and 3.5 Mbps upstream. G.992.3 ADSL2.
ADSL2plus	The modem negotiates rates up to 28 Mbps downstream and 1 Mbps upstream (Annex M allows upstream up to 3.5 Mbps) G.992.5 ADSL2+.
Autonegotiate	The modem automatically negotiates all supported transmission modes.
Full rate	Full rate T1 ADSL modem. This is used for connecting to full rate T1.413 issue 2 modems.
G.dmt	G.dmt is a higher-bandwidth variant of G.lite that provides for downstream speeds of up to 8160 Kbps. G.dmt is defined in ITU specification G.992.1.
G.lite	The modem negotiates rates up to 1536 Kbps upstream and 512 Kbps downstream (G.992.2).

ADSL2+ rate adaptation

The ADSL2+ cards support rate adaptation, which allow them to respond to changing line conditions by adjusting the line rate. At startup, ADSL2+ADSL2+ modems may negotiate a data rate. The **rateMode** parameter allows the selection of three types of rate adaption:

- **fixed:** rate is fixed at the max configured rate.
- **adaptatstartup:** rate is set to the best possible speed (between min and max) during training and does not change afterward.
- **adaptatruntime:** rate is set to the best possible speed (between min and max) during training and can change afterward based on changing conditions

The default option is **adaptatruntime**, so the rate can change based on changing conditions.

Advanced ADSL2+ configurations on the MXK

This section describes configuring signal-to-noise (SNR) parameters and describes:

- [Fine tuning ADSL2+ video performance, page513](#)
- [Seamless Rate Adaptation, page516](#)
- [Transport mode: fast or interleaved, page518](#)

ADSL2+ modems use signal-to-noise ratio (SNR) measurements to adjust signal transmission to achieve greater performance. The Zhone default

settings for SNR parameters normally provide an excellent throughput rate for most applications.

Fine tuning ADSL2+ video performance

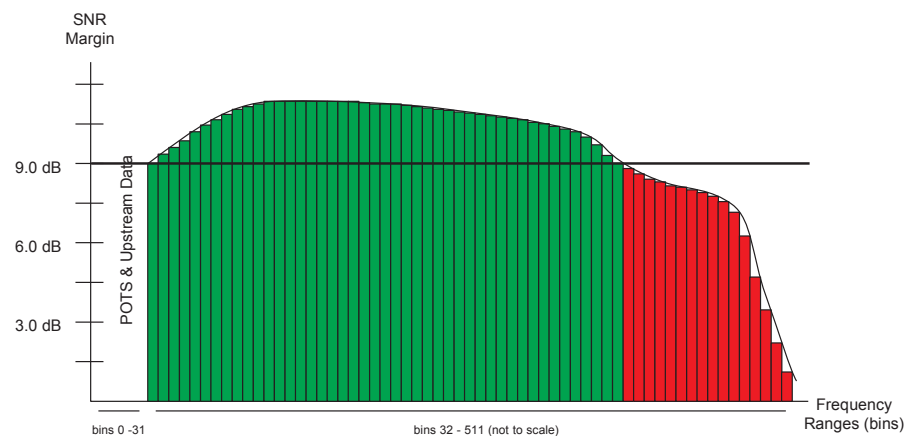
The parameters for tuning performance may be adjusted for video. These parameters are part of a complex system. Before making changes to the default settings you should understand the SNR parameters and how they work together.

This section describes guidelines for adjusting SNR settings and will not be correct for every deployed line. Subscribers with “noisy” lines may need to have their ADSL2+ parameters adjusted so that the train rates are high enough to meet the service bandwidth requirements. This section discusses how adjusting SNR Margins can increase train rates while keeping errors on the line to a minimum.

SNR compares the level of the desired signal to the level of background noise. The better the signal and the less obtrusive the background noise, the higher the ratio. The lower the SNR, the greater effect noise will have on the ADSL2+ signal. Noise is anything that will corrupt the sent signal and is typically from AM radio transmissions, although poor physical connections, deformities in the line, transformers, and even appliances can also introduce noise.

If it weren't for noise you could set the SNR very high and not be worried about signal degradation. Unfortunately in the field, not all ADSL2+ lines will train when the SNR margin target is set high. With the target SNR margin set too high, the ADSL2+ training algorithm would try to make the line so clean (no noise) that the train rate would be very low and not capable of supporting the services sold.

Figure 47: Bins shown with SNR Margin set to 9.0 dB



The frequency bands on DSL lines are segmented into small frequency ranges called bins or tones. These small ranges make it so the frequency can be sampled to judge the value. There are 512 bins in a signal. The voice and

upstream data traffic use only a small portion (bins 0-31) and are not relevant to this discussion. Bins 32-511 are used for downstream data traffic.

If the SNR is dropped to a lower rate with the same signal to noise ratio, more of the sampled bins are used.

Figure 48: Bins shown with SNR Margin set to 6.0 dB

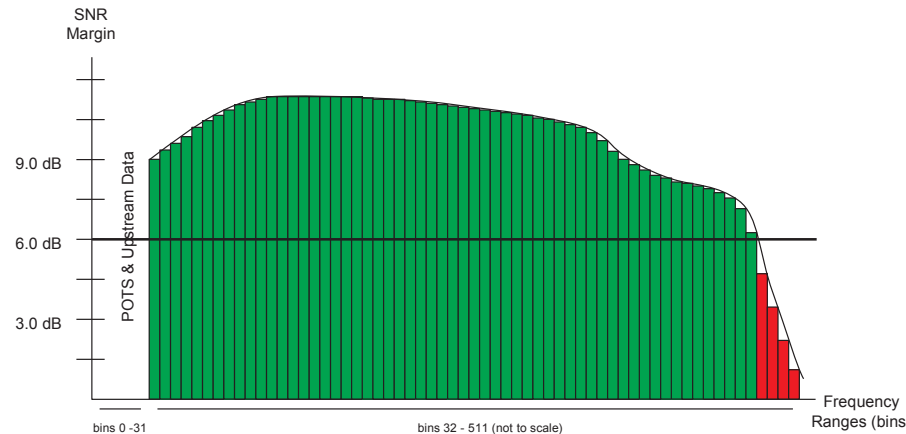


Figure47 and Figure48 show a snapshot of the signal.

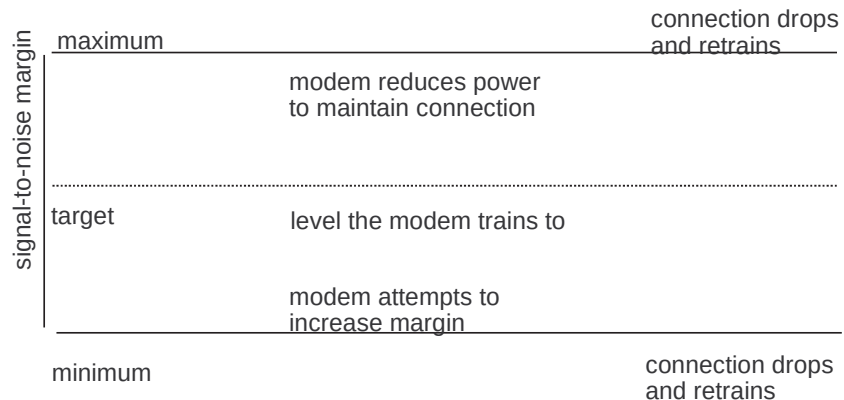
Table37 describes the three parameters in the **adsl-co-profile** and the **adsl-cpe-profile** which define the training speeds.

Table 37: adsl-co-profile and adsl-cpe-profile parameters

Parameter	Profile	Description
targetSnrMgn	adsl-co-profile adsl-cpe-profile	The Target SNR Margin (targetSnrMgn) is the SNR Margin targeted when training. Values are from 0 to 310 in tenths of dBs. A value of 60 would mean 6.0 dB SNR Default: 0 Recommendation for Video: 60
maxSnrMgn	adsl-co-profile adsl-cpe-profile	Maximum SNR Margin (maxSnrMgn) is the maximum SNR Margin allowed on the link before a retrain is forced. Values are from 0 to 310 in tenths of dBs. A value of 150 would mean 15.0 dB SNR. Default: 0 Recommendation for Video: 150
minSnrMgn	adsl-co-profile adsl-cpe-profile	Minimum SNR Margin (minSnrMgn) is the minimum SNR Margin allowed on the link before a retrain is forced. Values are from 0 to 310 in tenths of dBs. Default: 0 Recommendation for Video: 30

SNR performance is monitored to maintain a bit error rate (BER) of 10^{-7} or better. The minimum margin is the floor at which the modem will maintain a connection. The maximum margin is the ceiling for power cutback. The target margin is the lowest margin that the modem tries to achieve when training and adapting. [Figure 49](#) shows single-to-noise margin values.

Figure 49: Signal-to-noise margins



These three values alone allow the ADSL2+ line to train to a maximum rate given the target SNR Margin value. That initial train rate would remain unless the SNR Margin moves beyond the Minimum or Maximum SNR Margin. At that time the link is forced to retrain.

The system will try to attain the target signal-to-noise margin when training. If the line reaches the maximum bit rate and the actual margin is below the maximum margin, the line operates normally. If the margin rises above the target margin, the modem drops the connection and retrains once, then drops the power to enforce the maximum margin.

If, after a connection is made, the margin drops below the target margin, the modem attempts to increase the margin. If the minimum margin cannot be kept, the modem drops the connection and retrains.

Note within the above table are the Zhone recommended values for video. These SNR Margin values may not be appropriate on every link, but based on Zhone's testing they result in high train rates and low error rates on most lines. For loops with excessive noise which prevents the necessary data rate for video services, adjust the targetSnrMgn to 60. Lowering the Target SNR Margin should allow the line to train higher.

Retraining the signal takes a considerable amount of time (as much as 30 seconds). An ADSL2+ feature Seamless Rate Adaption (SRA) can make more minute adjustments within the minimum and maximum SNR margins without the end user being aware of the rate changes or time to retrain.

Seamless Rate Adaptation

After an ADSL2+ link trains the noise conditions on the line could improve. Seamless Rate Adaptation allows the ADSL2+ link to take advantage of the lower noise and will increase the rate of the link without the need for a retrain. SRA may also reduce the rate on the line when noise levels increase slightly.

The Upshift SNR Margin (**upshiftSnrMgn**) and Downshift SNR Margin (**downshiftSnrMgn**) are used to determine the values to conduct the rate adaptation by adding or removing bins to stay at the target SNR. Time parameters work with the Upshift and Downshift SNR Margins, Minimum Upshift SNR Time (**minUpshiftTime**) and Minimum Downshift Time (**minDownshiftTime**) for the Central Office (**adsl-co-profile**) and Minimum Upshift SNR Margin (**minUpshiftSnrMgn**) and Minimum Downshift SNR Margin (**minDownshiftSnrMgn**), which are also for the modem side of the connection (**adsl-cpe-profile**). The CO profile and CPE profile use different names for the similar parameter because the CO is the head of the connection and accepts requests from the CPE devices, but can determine the minimum time to conduct the SRA.

All of these parameters work together in a system. When the SNR rises above the Upshift SNR Margin and stays there for a specified amount of time (from the minUpshiftTime and minUpshiftSnrMgn) it is assumed that the noise level has improved and the rate is allowed to increase.

As the SNR moves below the Downshift SNR Margin value and stays there for a specified amount of time, the noise level has increased and the current noise level can not sustain the current downstream rate without increased errors so the rate is decreased. The increases and decreases in rate are done “seamlessly” and without the need to retrain the line.

Figure 50: SNR Margins working as a system

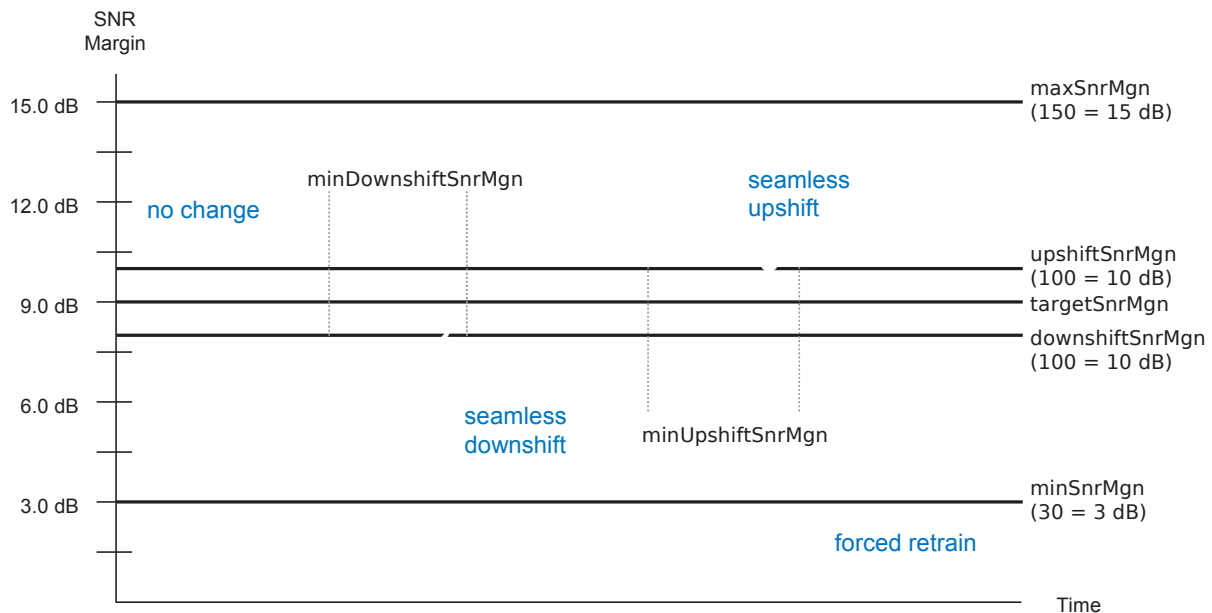


Figure 50 shows how the five SNR Margin parameters work as a system to ensure the best train rate possible within the given parameters. The red line represents how the SNR changes over time. The SNR Margin increases, but does not move past the Upshift SNR Margin at (1) so the train rate remains the same. At (2) on the graph the SNR Margin has dipped below the Downshift SNR Margin and stays below downshiftSnrMgn longer than the minimum downshift margin time. This situation results in a removal of bins in order to return to the Target SNR Margin. This change is a seamless decrease in the data rate from the user's perspective. The SNR Margin then rises and moves above the Upshift SNR Margin for longer than **minUpshiftSnrMgn** period resulting in a seamless increase in the rate at (3). In this situation bins are added to get back to the Target SNR Margin. The SNR then moves down quickly below the Min SNR Margin which forces a retrain at (4).

Note that each parameter plays an important role in the training of the ADSL2+ line. The SNR margins should always have **maxSnrMgn > upshiftSnrMgn > targetSnrMgn > downshiftSnrMgn > minSnrMgn**. If the Minimum and Maximum SNR Margins are brought too close to the target SNR Margin on a line which has changing SNR, there could be excessive retraining. If the SRA values Upshift SNR Margin and Downshift SNR Margin are too close to the Maximum and Minimum SNR values, SRA will not be useful, the line will retrain by the Minimum and Maximum SNR values.

Setting the SRA shift values too high for the upshift and too low for the downshift makes the probability of an SRA shift unlikely. A good configuration rule for determining **downshiftSnrMgn** and **upshiftSnrMgn**:

- $\text{downshiftSnrMgn} = \text{targetSnrMgn} + 10$

- $\text{upshiftSnrMgn} = \text{targetSnrMgn} - 10$

SRA is only supported in the downstream data direction and the CPE is the controlling device for the feature. SRA is configured in the **adsl-cpe-profile**. Changes to the **adsl-co-profile** are ignored.

There are two timers used to space SRA events. The downstream (CO to CPE) SRA timers are located in the **adsl-cpe-profile**. The SRA timers are in units of seconds so a value of 60 means an SRA event can only occur every 60 seconds.

Zhone's recommended settings are:

- $\text{minUpshiftSnrMgn} = 30$
- $\text{minDownshiftSnrMgn} = 30$

The SRA timers start after the first SRA action which means that an SRA rate shift can occur immediately after initial train up.

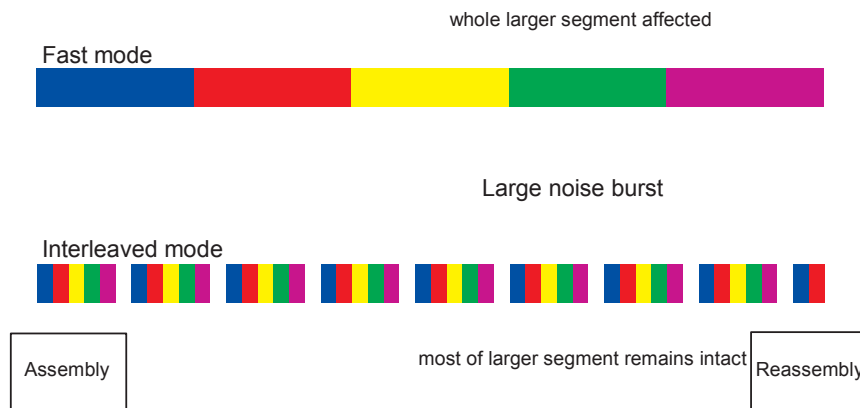
For SRA to operate the CPE must support SRA and must have SRA enabled.

Transport mode: fast or interleaved

ADSL2+ operates in one of two modes: fast or interleaved. In fast mode, data packets are placed on the ADSL2+ line contiguously providing data in a sequential stream. The other end of the ADSL2+ line removes the data packets in order, then moves them up the protocol stack for processing. In interleaved mode, the data packets are broken into smaller segments, then sent down the line.

The advantage of fast mode is that the data is streamed directly without disassembly and assembly processing. However, a short burst of noise on the line can corrupt more errors than the far end device can correct. ADSL2+ modems have the ability to correct errors; however error correction works well when there are a small number of errors to correct. Too many bit errors in a packet can mean the errors can not be corrected and result in lost data packets. Lost data packets require that the same data packet be retransmitted.

With smaller segments use interleaving mode, the segments may be intermixed with segments from other packets before being placed on the ADSL2+ line. If a burst of noise causes corruption, each entire larger pre-disassembled packet is not affected as much, but the smaller pieces which could belong to several different packets are affected. Because only smaller segments of the larger packet are affected, error correction is more likely to have enough information to build the packet correctly on reassembly. The data packets handed up the stack will have no issues.

Figure 51: Fast and interleaved mode

The drawback with Interleaving is that the process of interleaving the small data blocks and reassembling the data packets at the far end introduce some delay and lowers the data rate.

It is recommended to use Fast mode with data applications.

Interleaved mode should be used with video applications. Video applications usually do not support retransmissions. If a data packet is corrupted it is discarded and will not be retransmitted so it is important that as many packets as possible arrive in good condition.

Fast configuration notes

On the MXK, fast and interleave modes are configured in the **adslChannelMode** field in the **adsl-profile**.

[Table38](#) shows the settings are used for fast operations.

Table 38: Fast operation settings

Parameter	Profile	Description
fastMinTxRate	adsl-co-profile adsl-cpe-profile	Minimum transmit rate in bits per second (bps) for channels configured for fast transmission mode. fastMinTxRate must be less than fastMaxTxRate . Default: 0
fastMaxTxRate	adsl-co-profile adsl-cpe-profile	Configured maximum transmit rate (bps) for ADSL Fast channels. fastMaxTxRate must be greater than fastMinTxRate . Default: 8460Kbps

Table 38: Fast operation settings (Continued)

Parameter	Profile	Description
threshFastRateUp	adsl-co-profile adsl-cpe-profile	Not currently used. The change in the configured rate that causes the system to send an adslAtucRateChangeTrap. The system sends a trap whenever: ChanCurrTxRate <= ChanPrevTxRate minus the value of this object. A value of 0 disables the trap. Default: 0
threshFastRateDown	adsl-co-profile adsl-cpe-profile	Not currently used. The change in the configured rate that causes the system to send an adslAturRateChangeTrap. The system sends a trap whenever: ChanCurrTxRate <= ChanPrevTxRate minus the value of this parameter. Default: 0

Interleaved configuration notes

[Table39](#) shows the settings used during interleaved operations.

Table 39: Interleaved operation settings

Parameter	Profile	Description
maxInterleaveDelay	adsl-co-profile adsl-cpe-profile	Defines the mapping (relative spacing) between subsequent interleave input bytes and their placement in the bit stream at the interleave output. Larger numbers provide greater separation between consecutive input bytes in the output bit stream allowing for improved impulse noise immunity, but at the expense of payload latency. Default: 0
interleaveMinTxRate	adsl-co-profile adsl-cpe-profile	Minimum transmit rate (bps) for channels configured for interleaved transmission mode. interleaveMinTxRate must be less than interleaveMaxTxRate. Default: 0
interleaveMaxTxRate	adsl-co-profile adsl-cpe-profile	Maximum transmit rate (bps) for channels configured for interleaved transmission mode. Default: 8160Kbps

Table 39: Interleaved operation settings (Continued)

Parameter	Profile	Description
threshInterleaveRateUp	adsl-co-profile adsl-cpe-profile	Not currently used. The change in the configured rate that causes the system to send an <code>adslAturRateChangeTrap</code>. The system sends a trap whenever: ChanCurrTxRate $\geq$ ChanPrevTxRate plus the value of this object. Default: 0
threshInterleaveRateDown	adsl-co-profile adsl-cpe-profile	Not currently used. The change in the configured rate that causes the system to send an <code>adslAtucRateChangeTrap</code>. The system sends a trap whenever: ChanCurrTxRate $\leq$ ChanPrevTxRate minus the value of this object. Default: 0

Configure ADSL2+ interfaces

This section explains ADSL2+ configuration on the MXK. It contains the following sections:

- [ADSL2+ interface overview, page521](#)
- [View adsl-profile parameter defaults, page522](#)
- [View adsl-co-profile parameter defaults, page525](#)
- [View adsl-cpe-profile parameter defaults, page530](#)
- [Upstream and downstream tone ranges, page536](#)
- [Configure ADSL2+ profiles for Annex M in fast mode, page536](#)
- [Configure ADSL2+ profiles for Annex M in interleaved mode, page538](#)
- [Configure ADSL2+ profiles for G.lite, page541](#)
- [Configure ADSL2+ profiles to cap train rates, page543](#)
- [Configure ADSL2+ S=1/2, page548](#)
- [Configure Broadcom Phy-R™ parameters, page553](#)

ADSL2+ interface overview

ADSL2+ configuration consists of three profiles:

- **adsl-profile**
- **adsl-co-profile**
- **adsl-cpe-profile**

[Table40](#) summarizes the **update** commands used to configure ADSL2+ interfaces on the MXK:

Table 40: Commands to configure ADSL2+ interfaces

Action	Command
Configure the ADSL2+ interface. See Configure ADSL2+ profiles for Annex M in fast mode on page536 .	update adsl-profile shelf/slot/port where <i>port</i> is from 1 to 48
Configure the downstream interface. See Configure ADSL2+ profiles for Annex M in fast mode on page536 .	update adsl-co-profile shelf/slot/port
Configure the upstream interface. See Configure ADSL2+ profiles for Annex M in fast mode on page536 .	update adsl-cpe-profile shelf/slot/port
Configure ADSL S=1/2. See Configure ADSL2+ S=1/2 on page548 .	update adsl-profile shelf/slot/port update adsl-co-profile shelf/slot/port

View adsl-profile parameter defaults

The **adsl-profile** defaults are appropriate for most applications. When necessary, the **adsl-profile** parameters can be updated.

Viewing the adsl-profile defaults

View the **adsl-profile** parameters and their default values.

- 1 View the ADSL2+ cards:

```
zSH> slots
Uplinks
a:*MXK EIGHT GIGE (RUNNING)
b: MXK EIGHT GIGE (NOT_PROV)
Cards
1: MXK ADSL-48-A Bonded (RUNNING)
2: MXK ADSL-48-A Bonded (RUNNING)
11: MXK ADSL-48-A Bonded/with 900 Ohm Splitter (RUNNING)
17: MXK ADSL-48-A Bonded/with 600 Ohm Splitter (RUNNING)
```

- 2 View the **adsl-profile** default parameters and values:

```
zSH> show adsl-profile
adslLineConfProfile:-----> {260}
adslAlarmConfProfile:-----> {260}
adslTrellisModeEnabled:-----> true false
adslNTRModeEnabled:-----> true false
adslTransmissionMode:-----> autonegotiatemode fullratemode glitemode
t1mode gdmtdmode ghsmode adsl2mode adsl2plusmode reachdslmode
```

```

adslChannelMode:-----> fastonly interleavedonly fastandinterleaved
fastorinterleaved
adslMaxDownstreamToneIndex:-----> {6 - 1023}
adslMinDownstreamToneIndex:-----> {6 - 1023}
adslMaxUpstreamToneIndex:-----> {1 - 63}
adslMinUpstreamToneIndex:-----> {1 - 63}
adslPotsBypassRelayMaxDuration:-> {1 - 300}
adslLineDMTConfMode:-----> echocancel freqdivmux
adslAnnexMModeEnabled:-----> true false
adslAnnexMPsdMask:-----> eu64 eu60 eu56 eu52 eu48 eu44 eu40 eu36
eu32 all

```

[Table41](#) defines **adsl-profile** parameters values.

Table 41: adsl-profile parameter definitions

Parameter	Description
adslLineConfProfile	Read only.
adslAlarmConfProfile	Read-only.
adslTrellisModeEnabled	Enables or disables trellis mode.
adslTransmissionMode	ADSL transmission mode. Supported values: Values: adsl2mode The modem negotiates rates up to G.992.3 and G.992.4 ADSL2. ADSL2plusmode The modem negotiates rates up to G.992.5 (ADSL2+). autonegotiatemode : automatically negotiates all supported transmission modes. fullratemode : automatically negotiates full rate modes (G.dmt and T1 mode). G.dmt has priority over T1 mode. glitemode : G.lite. Supports only interleave mode. t1mode : Full rate T1 gdmtmode : G.dmt ghsmode : the modem negotiates only G.dmt and G.lite modes. G.dmt has priority over G.lite. reachdslmode the modem negotiates only reach DSL mode. Default: autonegotiatemode
adslChannelMode	Specifies the channelization of the ADSL line. Supported values: Values: fastonly No impulse noise protection, but lowest possible latency. Recommended only where lowest possible latency is required (for example, gaming) interleavedonly Better impulse noise protection with higher latency. Recommended for all voice, video, and/or data deployments. Default: fastonly

Table 41: adsl profile parameter definitions (Continued)

Parameter	Description
adslMaxDownstreamToneIndex	Specifies the maximum downstream active tone. Values: 32 (128KHz) to 511 (2044KHz) Each value represents 4KHz. Default: 255 Changing this value causes the DSL modems to retrain.
adslMinDownstreamToneIndex	Specifies the minimum downstream active tone. Values: 32 (128KHz) to 255 (1020KHz) Each value represents 4KHz. Default: 511 - Changing this value causes the DSL modems to retrain. - For Annex B and Annex M configurations, this value should be set to 64.
adslMaxUpstreamToneIndex	Specifies the maximum upstream active tone. Values: 6 (24KHz) to 30 (120KHz) Each value represents 4KHz. Default: 3' - Changing this value causes the DSL modems to retrain. - For Annex B and Annex M configurations, this value should be set to 63.
adslMinUpstreamToneIndex	Specifies the minimum upstream active tone. Values: 6 (24KHz) to 30 (120KHz) Each value represents 4KHz. Default: 6 Changing this value causes the DSL modems to retrain.
adslPotsBypassRelayMaxDuration	Not currently used. The maximum duration in seconds that an ADSL POTS low-pass filter bypass relay will remain active (closed). The relay will automatically return a line back to normal (open) mode when this timer has expired. Values: 1 to 300 Default: 60 Only valid for ADSL-SPLTR-32 cards.

Table 41: adsl-profile parameter definitions (Continued)

Parameter	Description
adslLineDMTConfMode	Selects whether there is overlap of ADSL Discrete Multi-Tone (DMT) frequency bins. Values: echoCancel overlap of DMT frequency bins. Only supported by g.dmt Annex A. freqDivMux no overlap of DMT frequency bins. Separates downstream and upstream transmissions. Default: freqdivmux
annexMModeEnabled	Specifies whether annex M mode is enabled. This parameter can only be set to true when the adslTransmissionMode parameter is set to autonegotiate, adsl2mode, or adsl2plusmode. Default: false
adslAnnexMPsdMask	eu64 eu60 eu56 eu52 eu48 eu44 eu40 eu36 eu32 all

View adsl-co-profile parameter defaults

The ADSL2+ downstream interface is **adsl-co-profile** which defines the downstream behavior.

Viewing the adsl-co-profile defaults

View the **adsl-co-profile** parameters and their default values.

- 1 View the ADSL2+ cards:

```

zSH> slots
Uplinks
a:*MXK EIGHT GIGE (RUNNING)
b: MXK EIGHT GIGE (NOT_PROV)
Cards
1: MXK ADSL-48-A Bonded (RUNNING)
2: MXK ADSL-48-A Bonded (RUNNING)
11: MXK ADSL-48-A Bonded/with 900 Ohm Splitter (RUNNING)
17: MXK ADSL-48-A Bonded/with 600 Ohm Splitter (RUNNING)

```

- 2 View the **adsl-co-profile** default parameter values:

```

zSH> get adsl-co-profile 1/17/1
adsl-co-profile 1/17/1
rateMode: -----> {adaptatruntime}
rateChanRatio: -----> {50}
targetSnrMgn: -----> {60}
maxSnrMgn: -----> {310}
minSnrMgn: -----> {0}
downshiftSnrMgn: -----> {30}
upshiftSnrMgn: -----> {90}
minUpshiftTime: -----> {60}
minDownshiftTime: -----> {60}

```

```

fastMinTxRate: -----> {32000}
interleaveMinTxRate: -----> {32000}
fastMaxTxRate: -----> {32736000}
maxInterleaveDelay: -----> {63}
interleaveMaxTxRate: -----> {32736000}
thresh15MinLofs: -----> {0}
thresh15MinLoss: -----> {0}
thresh15MinLols: -----> {0}
thresh15MinLprs: -----> {0}
thresh15MinESs: -----> {0}
threshFastRateUp: -----> {0}
threshInterleaveRateUp: ---> {0}
threshFastRateDown: -----> {0}
threshInterleaveRateDown: -> {0}
initFailureTrapEnable: ----> {disable}
reachExtendedAdsl2: -----> {enable}
minTxThresholdRateAlarm: --> {0}
minINP: -----> {20}
phyRSupport: -----> {disable}
phyRmaxINP: -----> {0}
phyRminRSoverhead: -----> {0}
phyRRtxRatio: -----> {0}
txPowerAttenuation: -----> {20}
cabMode: -----> {0}

```

[Table42](#) defines the values for the **adsl-co-profile** parameters.

:

Table 42: adsl-co-profile parameter definitions

Parameter	Description
rateMode	The transmit rate adaptation configured on this modem. Supported values: fixed: The rate is negotiated at startup and remains fixed. Modem speed is determined by the fastMaxTxRate or interleaveMaxTxRate parameters. adaptatstartup: The rate is negotiated at startup and remains fixed. Modem speed is determined by the fastMaxTxRate or interleaveMaxTxRate parameters. If the line is able to support a higher rate, the rate above the minimum is assigned to the available channel (either fast or interleave). adaptatruntime: The rate is negotiated dynamically and can vary between the maximum and minimum configured rates. If the line conditions change during runtime, the line speed is adjusted. Recommended for video. default: adaptatruntime
rateChanRatio	Configured allocation ratio of excess transmit bandwidth between fast and interleaved channels. default: 50

Table 42: adsl-co-profile parameter definitions (Continued)

Parameter	Description
targetSnrMgn	Target signal to noise margin (in tenths of dBs). This is the noise margin the modem must achieve with a BER of 10^{-7} or better to successfully complete initialization. Suggested values are 6 dB for data-only or data-voice service and 10 dB for video service with better protection against noise which causes tiling. default: 60
maxSnrMgn	Maximum acceptable signal/noise margin (in tenths of dBs). If the noise margin rises above this the modem attempts to reduce its power output to optimize its operation. Reduces crosstalk into other ADSL circuits by not transmitting at an unnecessarily high level. For video, suggested values are 31 for both upstream and downstream. default: 310
minSnrMgn	Minimum acceptable signal to noise margin (in tenths of dBs). If the noise margin falls below this level, the modem attempts to increase its power output. If that is not possible the modem will attempt to re-initialize or shut down. For video, use 2 downstream and 0 upstream and adjust downstream rate proactively just before video degrades. default: 0
downshiftSnrMgn	Configured Signal/Noise Margin for rate downshift. If the noise margin falls below this level, the modem should attempt to decrease its transmit rate. default: 0
upshiftSnrMgn	Configured Signal/Noise Margin for rate upshift. If the noise margin rises above this level, the modem should attempt to increase its transmit rate. default: 0
minUpshiftTime	Minimum time that the current margin is above UpshiftSnrMgn before an upshift occurs. default: 0
minDownshiftTime	Minimum time that the current margin is below DownshiftSnrMgn before a downshift occurs. default: 0
fastMinTxRate	Minimum transmit rate (in bps) for channels configured for fast transmission mode. For a CO interface, the range is 32Kbps to 8160Kbps (1536Kbps for G.Lite). default: 32 Kbps

Table 42: adsl-co-profile parameter definitions (Continued)

Parameter	Description
interleaveMinTxRate	Minimum transmit rate (in bps) for channels configured for interleaved transmission mode. For a CO interface, the range is 32Kbps to 8160Kbps (1536Kbps for G.Lite). default: 32 Kbps
fastMaxTxRate	Maximum transmit rate (in bps) for channels configured for fast transmission mode. For a CO interface, the range is 32Kbps to 8160Kbps (1536Kbps for G.Lite). default: 32 Kbps
maxInterleaveDelay	Maximum interleave delay for this channel. Interleave delay applies only to the interleave channel and defines the mapping (relative spacing) between subsequent input bytes at the interleave input and their placement in the bit stream at the interleave output. Larger numbers provide greater separation between consecutive input bytes in the output bit stream allowing for improved impulse noise immunity, but at the expense of payload latency. For video, to maximize protection of downstream signal (where impulse problems occur), minimize round-trip latency by minimizing upstream delay use 1 ms upstream and 16 ms downstream. Values: 0 0.5 ms 1 1 ms 2 2 ms 4 4 ms 8 8 ms 16 16 ms 32 32 ms 63 63 ms default: 63 ms
interleaveMaxTxRate	Maximum transmit rate (in bps) for channels configured for interleaved transmission mode. For a CO interface, the range is 32Kbps to 8160Kbps (1536Kbps for G.Lite). default: 32 Mbps
thresh15MinLofs	The number of Loss of Frame Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfLofsThreshTrap. default: 0

Table 42: adsl-co-profile parameter definitions (Continued)

Parameter	Description
thresh15MinLoss	The number of Loss of Signal Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfLossThreshTrap. default: 0
thresh15MinLols	The number of Loss of Link Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfLolsThreshTrap. default: 0
thresh15MinLprs	Not currently used. The number of Loss of Power Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfLprsThreshTrap. default: 0
thresh15MinESs	The number of Errored Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfESsThreshTrap. default: 0
threshFastRateUp	Not currently used. Applies to 'Fast' channels only. Configured change in rate causing an adslAtucRateChangeTrap. default: 0
threshInterleaveRateUp	Not currently used. For 'Interleave' channels only. Configured change in rate causing an adslAtucRateChangeTrap. default: 0
threshFastRateDown	Not currently used. For 'Fast' channels only. Configured change in rate causing an adslAtucRateChangeTrap. default: 0
threshInterleaveRateDown	Not currently used. For 'Interleave' channels only. Configured change in rate causing an adslAtucRateChangeTrap. default: 0
initFailureTrapEnable	Not currently used. Enables and disables the InitFailureTrap. This trap controls whether line up or line down traps are sent while the system is booting up. default: 0

Table 42: adsl-co-profile parameter definitions (Continued)

Parameter	Description
reachextendedAdsl2	Defines whether downstream reach extended ADSL2 (READSL2) operations should be enforced by the ATU-C. Only enabled for ADSL2 and ADSL2+. Values: enable disable default: enable
minTxThresholdRateAlarm	Enables the CO (downstream) transmission rate threshold value. If the rate falls below this value, the device sends a trap and an alarm. default: 0
minINP (already used in the case of normal interleaving)	This parameter (already used in the case of normal interleaving) defines the minimal guaranteed impulse noise protection, provided that the available data bandwidth allowed for retransmissions is not exceeded.
phyRSupport	Enable to turn on Phy-R™ parameters. Disable to turn off Phy-R™ parameters. default: 0
phyRmaxINP	This parameter defines the maximum number of consecutive retransmissions that may take place and therefore bounds the maximal jitter due to retransmissions. A default value of zero doesn't bound the number of consecutive retransmissions (that will however never exceed maxDelay * 4 symbols).
minRSoverhead	This new parameter allows to force a minimum amount of RS overhead. This can be used to guarantee a given amount of steady state error correction capability. A default of zero doesn't force the use of RS overhead. default: 0
phyRRtxRatio	This parameter allows to provision a minimal guaranteed retransmission bandwidth, on top of the minimum rate. In case of the repetitive impulses of known maximal length and periodicity, this parameter can be used to guarantee that the repetitive impulse noise can be corrected. A default of zero doesn't force any extra guaranteed data bandwidth for retransmissions. default: 0
txPowerAttenuation	
cabMode	

View adsl-cpe-profile parameter defaults

The ADSL2+ upstream interface is **adsl-cpe-profile** which defines the upstream behavior.

Viewing the adsl-cpe-profile defaults

View the **adsl-cpe-profile** parameters and their default values.

1 View the ADSL2+ cards.

```
zSH> slots
Uplinks
  a:*MXK EIGHT GIGE (RUNNING)
  b: MXK EIGHT GIGE (NOT_PROV)
Cards
  1: MXK ADSL-48-A Bonded (RUNNING)
  2: MXK ADSL-48-A Bonded (RUNNING)
 11: MXK ADSL-48-A Bonded/with 900 Ohm Splitter (RUNNING)
 17: MXK ADSL-48-A Bonded/with 600 Ohm Splitter (RUNNING)
```

2 View the default parameters for the **adsl-cpe-profile**:

```
zSH> get adsl-cpe-profile 1/17/1
adsl-cpe-profile 1/17/1
rateMode: -----> {adaptatruntime}
rateChanRatio: -----> {50}
targetSnrMgn: -----> {60}
maxSnrMgn: -----> {310}
minSnrMgn: -----> {0}
downshiftSnrMgn: -----> {30}
upshiftSnrMgn: -----> {90}
minUpshiftSnrMgn: -----> {60}
minDownshiftSnrMgn: -----> {60}
fastMinTxRate: -----> {32000}
interleaveMinTxRate: -----> {32000}
fastMaxTxRate: -----> {1024000}
interleaveMaxTxRate: -----> {1536000}
maxInterleaveDelay: -----> {16}
thresh15MinLofs: -----> {0}
thresh15MinLoss: -----> {0}
thresh15MinLprs: -----> {0}
thresh15MinESs: -----> {0}
threshFastRateUp: -----> {0}
threshInterleaveRateUp: ---> {0}
threshFastRateDown: -----> {0}
threshInterleaveRateDown: -> {0}
minTxThresholdRateAlarm: --> {0}
minINP: -----> {0}
phyRSupport: -----> {disable}
phyRmaxINP: -----> {0}
phyRminRSoverhead: -----> {0}
phyRRtxRatio: -----> {0}
```

[Table43](#) defines the values for the **adsl-cpe-profile** parameters

:

Table 43: adsl-cpe-profile parameter definitions

Parameter	Description
rateMode	The transmit rate adaptation configured on this modem. Supported values: fixed: The rate is negotiated at startup and remains fixed. Modem speed is determined by the fastMaxTxRate or interleaveMaxTxRate parameters. adaptatstartup: The rate is negotiated at startup and remains fixed. Modem speed is determined by the fastMaxTxRate or interleaveMaxTxRate parameters. If the line is able to support a higher rate, the rate above the minimum is assigned to the available channel (either fast or interleave). adaptatruntime: The rate is negotiated dynamically and can vary between the maximum and minimum configured rates. If the line conditions change during runtime, the line speed is adjusted. default: adaptatruntime
rateChanRatio	Configured allocation ratio of excess transmit bandwidth between fast and interleaved channels. default: 50
targetSnrMgn	Target signal to noise margin (in tenths of dBs). This is the noise margin the modem must achieve with a BER of 10^{-7} or better to successfully complete initialization. default: 60
maxSnrMgn	Maximum acceptable signal/noise margin (in tenths of dBs). If the noise margin rises above this the modem attempts to reduce its power output to optimize its operation. default: 310
minSnrMgn	Configured Signal/Noise Margin for rate downshift. If the noise margin falls below this level, the modem should attempt to decrease its transmit rate. default: 0
downshiftSnrMgn	Configured Signal/Noise Margin for rate downshift. If the noise margin falls below this level, the modem should attempt to decrease its transmit rate. default: 0
upshiftSnrMgn	Minimum time that the current margin is above upshiftSnrMgn before an upshift occurs. default: 0
minUpshiftSnrMgn	Minimum time that the current margin is below. DownshiftSnrMgn before a downshift occurs. default: 0

Table 43: adsl-cpe-profile parameter definitions (Continued)

Parameter	Description
minDownshiftSnrMgn	Configured Signal/Noise Margin for rate downshift. If the noise margin falls below this level, the modem should attempt to decrease it's transmit rate. default: 0
fastMinTxRate	Minimum transmit rate (in bps) for channels configured for fast transmission mode. For a CPE interface, the range is 32 Kbps to 896 Kbps (512 for G.lite). default: 32 Kbps
interleaveMinTxRate	Minimum transmit rate (in bps) for channels configured for interleaved transmission mode. For a CPE interface, the range is 32Kbps to 896Kbps (1512Kbps for G.Lite). Default: 32 Kbps
fastMaxTxRate	Maximum transmit rate (in bps) for channels configured for fast transmission mode. For a CPE interface, the range is 32Kbps to 1024Kbps (512Kbps for G.Lite). Default: 1024 Kbps
interleaveMaxTxRate	Maximum transmit rate (in bps) for channels configured for interleaved transmission mode. For a CPE interface, the range is 32 Kbps to 1536Kbps (512 Kbps for G.lite). Default: 1536 Kbps

Table 42: adsl-cpe-profile parameter definitions (Continued)

Parameter	Description
maxInterleaveDelay	Maximum interleave delay for this channel. Interleave delay applies only to the interleave channel and defines the mapping (relative spacing) between subsequent input bytes at the interleave input and their placement in the bit stream at the interleave output. Larger numbers provide greater separation between consecutive input bytes in the output bit stream allowing for improved impulse noise immunity, but at the expense of payload latency. For video, to maximize protection of downstream signal (where impulse problems occur), minimize round-trip latency by minimizing upstream delay use 1 ms upstream and 16 ms downstream. Values: 0 0.5 ms 1 1 ms 2 2 ms 4 4 ms 8 8 ms 16 16 ms 32 32 ms 63 63 ms Default: 63 ms
thresh15MinLofs	The number of Loss of Frame Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfLofsThreshTrap. Default: 0
thresh15MinLoss	The number of Loss of Signal Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfLossThreshTrap. Default: 0
thresh15MinLprs	The number of Loss of Power Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfLprsThreshTrap. Default: 0
thresh15MinESs	The number of Errored Seconds encountered by an ADSL interface within any given 15 minutes performance data collection period, which causes the SNMP agent to send an adslAtucPerfESsThreshTrap. Default: 0
threshFastRateUp	Applies to 'Fast' channels only. Configured change in rate causing an adslAtucRateChangeTrap. Default: 0

Table 43: adsl-cpe-profile parameter definitions (Continued)

Parameter	Description
threshInterleaveRateUp	For 'Interleave' channels only. Configured change in rate causing an adslAtucRateChangeTrap. Default: 0
threshFastRateDown	For 'Fast' channels only. Configured change in rate causing an adslAtucRateChangeTrap. Default: 0
threshInterleaveRateDown	For 'Interleave' channels only. Configured change in rate causing an adslAtucRateChangeTrap. Default: 0
minTxThresholdRateAlarm	Enables the CO (downstream) transmission rate threshold value. If the rate falls below this value, the device sends a trap and an alarm. Default: 0
minINP (already used in the case of normal interleaving)	This parameter (already used in the case of normal interleaving) defines the minimal guaranteed impulse noise protection, provided that the available data bandwidth allowed for retransmissions is not exceeded.
phyRSupport	Enable to turn on Phy-R™ parameters. Disable to turn off Phy-R™ parameters. default: disable
phyRmaxINP	This parameter defines the maximum number of consecutive retransmissions that may take place and therefore bounds the maximal jitter due to retransmissions. A default value of zero doesn't bound the number of consecutive retransmissions (that will however never exceed maxDelay * 4 symbols). default: 0
phyRminRSoverhead	This new parameter allows to force a minimum amount of RS overhead. This can be used to guarantee a given amount of steady state error correction capability. A default of zero doesn't force the use of RS overhead. default: 0
phyRRtxRatio	This parameter allows to provision a minimal guaranteed retransmission bandwidth, on top of the minimum rate. In case of the repetitive impulses of known maximal length and periodicity, this parameter can be used to guarantee that the repetitive impulse noise can be corrected. A default of zero doesn't force any extra guaranteed data bandwidth for retransmissions. default: 0

Upstream and downstream tone ranges

The MXK supports setting the active upstream and downstream tone ranges for ADSL2+ modems. Since this is not usually required, understand that changing the range of tones can affect the maximum throughput of the channel as well as providing isolation from certain interference.

The following parameters in the **adsl-profile** specify the range of active tones for the DSL modem:

- **AdslMaxDownstreamToneIndex**
- **AdslMinDownstreamToneIndex**
- **AdslMaxUpstreamToneIndex**
- **AdslMinUpstreamToneIndex**

 **Note:** Changing of any of these parameters causes the modem to retrain.

Configure ADSL2+ profiles for Annex M in fast mode

Enabling Annex M enables more tones to the upstream.

 **Note:** If annex M is disabled, these values should be reset.

Configuring ASDL for Annex M in fastonly mode

[Table44](#) describes the profiles and parameters and suggested values to enable Annex M in fast mode.

Table 44: Profiles and parameters used to configure ADSL2+ for Annex M in fast mode

Profile	Parameter and value
adsl-profile	adslChannelMode: <i>fastonly</i> adslMinDownstreamToneIndex: 64 adslMaxUpstreamToneIndex: 63 adslAnnexMMode: <i>true</i>
adsl-co-profile (optional)	fastMaxTxRate: 16384000 (16 Mb)
adsl-cpe-profile	fastMaxTxRate: 3072000 (3 Mb)

1 Update the **adsl-profile** for Annex M fast mode:

```
zSH> update adsl-profile 1/9/1
adsl-profile 1/9/1
Please provide the following: [q]uit.
adslLineConfProfile: -----> {0000000386}: ** read-only **
adslAlarmConfProfile: -----> {0000000386}: ** read-only **
```



```

adslTrellisModeEnabled: -----> {true}:
adslNTRModeEnabled: -----> {false}:
adslTransmissionMode: -----> {autonegotiatemode}:
adslChannelMode: -----> {interleavedonly}: fastonly
adslMaxDownstreamToneIndex: -----> {511}:
adslMinDownstreamToneIndex: -----> {32}: 64
adslMaxUpstreamToneIndex: -----> {31}: 63
adslMinUpstreamToneIndex: -----> {6}:
adslPotsBypassRelayMaxDuration: -> {60}:
adslLineDMTConfMode: -----> {freqdivmux}:
adslAnnexMModeEnabled: -----> {false}: true
adslAnnexMPsdMask: -----> {eu56}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

2 Update the **adsl-co-profile** downstream interface and set the **fastMaxTxRate** to 16 Mb. (Optional)

```

zSH> update adsl-co-profile 1/9/1
adsl-co-profile 1/9/1
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {0}:
upshiftSnrMgn: -----> {90}:
minUpshiftTime: -----> {0}:
minDownshiftTime: -----> {0}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {32736000}: 16384000
maxInterleaveDelay: -----> {63}:
interleaveMaxTxRate: -----> {32736000}:
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLols: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
initFailureTrapEnable: ----> {disable}:
reachExtendedAdsl2: -----> {enable}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
txPowerAttenuation: -----> {20}:

```

```

cabMode: -----> {0}
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

3 Update the **adsl-cpe-profile** upstream interface and set the **fastMaxTxRate** to 3 Mb.

```

zSH> update adsl-cpe-profile 1/9/1
adsl-cpe-profile 1/9/1
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {30}:
upshiftSnrMgn: -----> {90}:
minUpshiftSnrMgn: -----> {60}:
minDownshiftSnrMgn: -----> {60}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {1024000}: 3072000
interleaveMaxTxRate: -----> {1536000}:
maxInterleaveDelay: -----> {16}:
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Configure ADSL2+ profiles for Annex M in interleaved mode

Enabling Annex M enables more tones to the upstream.



Note: If Annex M is disabled, these values should be reset.

Configuring ADSL2+ for Annex M in interleaved mode

Table 45 describes the profiles and parameters and suggested values to enable Annex M in interleaved mode.

Table 45: Profiles and parameters used to configure ADSL2+ for Annex M interleaved mode

Profile	Parameter and value
adsl-profile	adslChannelMode: <i>interleavedonly</i> adslMinDownstreamToneIndex: 64 adslMaxUpstreamToneIndex: 63 adslAnnexMMode: <i>true</i>
adsl-co-profile (optional)	interleaveMaxTxRate: 16384000 (16 Mb)
adsl-cpe-profile	interleaveMaxTxRate: 3072000 (3 Mb)

1 Update the **adsl-profile** for Annex M interleaved mode.

```

zSH> update adsl-profile 1/9/2
adsl-profile 1/9/2
Please provide the following: [q]uit.
adslLineConfProfile: -----> {0000000390}: ** read-only **
adslAlarmConfProfile: -----> {0000000390}: ** read-only **
adslTrellisModeEnabled: -----> {true}:
adslINTRModeEnabled: -----> {false}:
adslTransmissionMode: -----> {autonegotiatemode}:
adslChannelMode: -----> {fastonly}: interleavedonly
adslMaxDownstreamToneIndex: -----> {511}:
adslMinDownstreamToneIndex: -----> {32}: 64
adslMaxUpstreamToneIndex: -----> {31}: 63
adslMinUpstreamToneIndex: -----> {6}:
adslPotsBypassRelayMaxDuration: -> {60}:
adslLineDMTConfMode: -----> {freqdivmux}:
adslAnnexMModeEnabled: -----> {false}: true
adslAnnexMPsdMask: -----> {eu56}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

2 Update the **adsl-co-profile** downstream interface and set the **interleaveMaxTxRate** to 16 Mb. (Optional)

```

zSH> update adsl-co-profile 1/9/2
adsl-co-profile 1/9/2
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {0}:
upshiftSnrMgn: -----> {90}:

```

```

minUpshiftTime: -----> {0}:
minDownshiftTime: -----> {0}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {32736000}:
maxInterleaveDelay: -----> {63}:
interleaveMaxTxRate: -----> {32736000}: 16384000
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLols: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
initFailureTrapEnable: ----> {disable}:
reachExtendedAdsl2: -----> {enable}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
txPowerAttenuation: -----> {20}:
cabMode: -----> {0}
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

3 Update the **adsl-cpe-profile** upstream interface and set the **interleaveMaxTxRate** to 3 Mb.

```

zSH> update adsl-cpe-profile 1/9/2
adsl-cpe-profile 1/9/2
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {30}:
upshiftSnrMgn: -----> {90}:
minUpshiftSnrMgn: -----> {60}:
minDownshiftSnrMgn: -----> {60}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {1024000}:
interleaveMaxTxRate: -----> {1536000}: 3072000
maxInterleaveDelay: -----> {16}:
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:

```

```

threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Configure ADSL2+ profiles for G.lite

Configuring ADSL 2+ profiles for G.lite

When the transmission mode is set to G.lite, the channel mode must be set to interleaved.

[Table 46](#) describes the profiles and parameters and suggested values to enable G.lite mode.

Table 46: Profile and parameters used to configure ADSL2+ for G.lite

Profile	Parameter and value
adsl-profile	adslTransmissionMode: <i>glitemode</i> adslChannelMode: <i>interleavedonly</i>
adsl-co-profile	interleaveMaxTxRate: <i>1536000 kbps</i>
adsl-cpe-profile	interleaveMaxTxRate: <i>512000 kbps</i>

- 1 If you change the transmission rate to *glitemode* and the channel mode to *interleavedonly* in the **adsl-profile** and save the change, you may see this error message:

```

.....
Save changes? [s]ave, [c]hange or [q]uit: s
interleaveMaxTxRate set too high in ADSL C0 and CPE profiles to select
glitemode.

```

In this case, configure the **interleaveMaxTxRate** parameter in the **adsl-co-profile** and the **adsl-cpe-profile** to a correct value, then configure the **adsl-profile** for G.lite.

A valid value for the **adsl-co-profile interleaveMaxTxRate** is 1536Kbps or less.

Set the **interleaveMaxTxRate** in the **adsl-co-profile**:

```
zSH> update adsl-co-profile 1/9/5
```

```

adsl-co-profile 1/9/5
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {0}:
upshiftSnrMgn: -----> {90}:
minUpshiftTime: -----> {0}:
minDownshiftTime: -----> {0}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {32736000}:
maxInterleaveDelay: -----> {63}:
interleaveMaxTxRate: -----> {32736000}: 1536000
thresh15MinLoFs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLols: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
initFailureTrapEnable: ----> {disable}:
reachExtendedAdsl2: -----> {enable}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
txPowerAttenuation: -----> {20}:
cabMode: -----> {0}
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

- 2** A valid value for the **adsl-cpe-profile interleaveMaxTxRate** is 512 Kbps or less.

Set the **interleaveMaxTxRate** in the **adsl-cpe-profile**

```

zSH> update adsl-cpe-profile 1/9/5
adsl-cpe-profile 1/9/5
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {30}:
upshiftSnrMgn: -----> {90}:

```

```

minUpshiftSnrMgn: -----> {60}:
minDownshiftSnrMgn: -----> {60}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {1024000}:
interleaveMaxTxRate: -----> {1536000}:  512000
maxInterleaveDelay: -----> {16}:
thresh15MinLoFs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

3 Update the **adsl-profile** for G.lite:

```

zSH> update adsl-profile 1/9/5
adsl-profile 1/9/5
Please provide the following: [q]uit.
adslLineConfProfile: -----> {0000000402}:  ** read-only **
adslAlarmConfProfile: -----> {0000000402}:  ** read-only **
adslTrellisModeEnabled: -----> {true}:
adslNTRModeEnabled: -----> {false}:
adslTransmissionMode: -----> {autonegotiatemode}:  glitemode
adslChannelMode: -----> {fastonly}:  interleavedonly
adslMaxDownstreamToneIndex: -----> {511}:
adslMinDownstreamToneIndex: -----> {32}:
adslMaxUpstreamToneIndex: -----> {31}:
adslMinUpstreamToneIndex: -----> {6}:
adslPotsBypassRelayMaxDuration: -> {60}:
adslLineDMTConfMode: -----> {freqdivmux}:
adslAnnexMModeEnabled: -----> {false}:
adslAnnexMPsdMask: -----> {eu56}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Configure ADSL2+ profiles to cap train rates

This section provides typical examples of capping upstream and downstream train rates for both fast and interleaved mode.

Table 47 describes the profiles and parameters and suggested upstream and downstream train rates.

Table 47: Profiles and parameters for capping upstream and downstream train rates

Profile	Parameter and train rates
adsl-profile	adslChannelMode: <i>fastonly</i> or <i>interleavedonly</i>
adsl-co-profile	fastMaxTxRate: 20,000,000 bps or interleaveMaxTxRate: 20,000,000 bps
adsl-cpe-profile	Note: bps show some typical train rates for the upstream. fastTaxTxRate: 384,000 bps or interleaveMaxTxRate: 512,000 bps

Capping upstream and downstream train rates (fast mode)

1 Change the channel mode in the **adsl-profile** to *fastonly*:

```

zSH> update adsl-profile 1/9/10
adsl-profile 1/9/10
Please provide the following: [q]uit.
adslLineConfProfile: -----> {0000000422}: ** read-only **
adslAlarmConfProfile: -----> {0000000422}: ** read-only **
adslTrellisModeEnabled: -----> {true}:
adslINTRModeEnabled: -----> {false}:
adslTransmissionMode: -----> {autonegotiatemode}:
adslChannelMode: -----> {interleavedonly}: fastonly
adslMaxDownstreamToneIndex: -----> {511}:
adslMinDownstreamToneIndex: -----> {32}:
adslMaxUpstreamToneIndex: -----> {31}:
adslMinUpstreamToneIndex: -----> {6}:
adslPotsBypassRelayMaxDuration: -> {60}:
adslLineDMTConfMode: -----> {freqdivmux}:
adslAnnexMModeEnabled: -----> {false}:
adslAnnexMPsdMask: -----> {eu56}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

2 Cap the downstream rate:

```

zSH> update adsl-co-profile 1/9/10
adsl-co-profile 1/9/10
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:

```



```

downshiftSnrMgn: -----> {0}:
upshiftSnrMgn: -----> {90}:
minUpshiftTime: -----> {0}:
minDownshiftTime: -----> {0}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {32736000}: 20000000
maxInterleaveDelay: -----> {63}:
interleaveMaxTxRate: -----> {32736000}:
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLols: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ----> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
initFailureTrapEnable: ----> {disable}:
reachExtendedAdsl2: -----> {enable}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
txPowerAttenuation: -----> {20}:
cabMode: -----> {0}
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

3 Cap the upstream rate:

```

zSH> update adsl-cpe-profile 1/9/10
adsl-cpe-profile 1/9/10
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {30}:
upshiftSnrMgn: -----> {90}:
minUpshiftSnrMgn: -----> {60}:
minDownshiftSnrMgn: -----> {60}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {1024000}: 384000
interleaveMaxTxRate: -----> {1536000}:
maxInterleaveDelay: -----> {16}:
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLprs: -----> {0}:

```

```

thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Capping upstream and downstream train rates (interleaved mode)

1 Change the channel mode in the **adsl-profile** to *interleavedonly*:

```

zSH> update adsl-profile 1/9/11
adsl-profile 1/9/11
Please provide the following: [q]uit.
adslLineConfProfile: -----> {0000000426}: ** read-only **
adslAlarmConfProfile: -----> {0000000426}: ** read-only **
adslTrellisModeEnabled: -----> {true}:
adslNTRModeEnabled: -----> {false}:
adslTransmissionMode: -----> {autonegotiatemode}:
adslChannelMode: -----> {fastonly}: interleavedonly
adslMaxDownstreamToneIndex: -----> {511}:
adslMinDownstreamToneIndex: -----> {32}:
adslMaxUpstreamToneIndex: -----> {31}:
adslMinUpstreamToneIndex: -----> {6}:
adslPotsBypassRelayMaxDuration: -> {60}:
adslLineDMTConfMode: -----> {freqdivmux}:
adslAnnexMModeEnabled: -----> {false}:
adslAnnexMPsdMask: -----> {eu56}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

2 Cap the downstream rate:

```

zSH> update adsl-co-profile 1/9/11
adsl-co-profile 1/9/11
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {0}:
upshiftSnrMgn: -----> {90}:
minUpshiftTime: -----> {0}:

```

```

minDownshiftTime: -----> {0}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {32736000}: 20000000
maxInterleaveDelay: -----> {63}:
interleaveMaxTxRate: -----> {32736000}:
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLols: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
initFailureTrapEnable: ----> {disable}:
reachExtendedAdsl2: -----> {enable}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
txPowerAttenuation: -----> {20}:
cabMode: -----> {0}

```

```

.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

3 Cap the upstream rate:

```

zSH> update adsl-cpe-profile 1/9/11
adsl-cpe-profile 1/9/11
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {30}:
upshiftSnrMgn: -----> {90}:
minUpshiftSnrMgn: -----> {60}:
minDownshiftSnrMgn: -----> {60}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {1024000}:
interleaveMaxTxRate: -----> {1536000}: 512000 bps
maxInterleaveDelay: -----> {16}:
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:

```

```

threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Configure ADSL2+ S=1/2

This section describes S=1/2 mode transmission on ADSL2+ line cards.

The ADSL2+ S=1/2 specification, as defined in the ITU standard G.992.2, is a transmission mode that supports downstream data rates up to 12 Mbps at distances of 6,000 feet or less. ADSL2+=1/2 is configured in either fast mode or interleaved mode. See [Configuring S=1/2 transmission mode for fast mode on page550](#) and [Configuring S=1/2 transmission mode for interleaved mode on page551](#).

Configure interleaved channels in the **adsl-profile**:

Table 48: adsl-profile

adsl-profile	Description Both ATU-C and ATU-R	MXK support	Range supported	Default
adslLineConfProfile	Read-Only	PARTIAL	260 only	260
adslAlarmConfProfile	Read-Only	PARTIAL	261 only	260
adslTrellisModeEnabled	Enable/Disable Trellis Mode	Yes	Enable=TRUE/ Disable=FALSE	TRUE
adslINTRModeEnabled	Network Timing Recovery	Not Used	Enable=TRUE/ Disable=FALSE	TRUE
adslTransmissionMode	Sets Transmission Mode	Yes	Yes	Yes
adslChannelMode	Specifies Channelization (Fast/Interleave)	Per line (not per channel)	fastonly interleavedonly	fastonly
adslMaxDownstreamToneIndex	Maximum Downstream Active Tones	Yes	6 to 1023	511
adslMinDownstreamToneIndex	Minimum Downstream Active Tones	Yes	6 to 1023	32
adslMaxUpstreamToneIndex	Maximum Upstream Active Tones	Yes	1 to 63	31
adslMinUpstreamToneIndex	Minimum Upstream Active Tones	Yes	1 to 63	6

Table 48: adsl-profile (Continued)

adsl-profile	Description Both ATU-C and ATU-R	MXK support	Range supported	Default
adslPotsBypassRelayMaxDuration	Not Used	Not Used	1 to 300	Not Used
adslLineDMTConfMode	DMT Mode - Echo Cancel or Freq Div Mux	Freq Div Mux Only	Freq Div Mux Only	Freq Div Mux
adslAnnexMModeEnabled	Enable/Disable Annex-M	Yes	Enable=TRUE/ Disable=FALSE	FALSE
adslAnnexMPsdMask				

Set the maximum transmit rate in the **adsl-co-profile**:

Table 49: adsl-co-profile

adsl-co-profile	ATU-C	SLMS Supported	SLMS Range supported	SLMS Default
rateMode	Transmit Rate Adaptation	Yes	AdaptAtRuntime Only	AdaptAtRuntime
rateChanRatio	Ratio of avail versus min rates	Defaulted	0 to 100	Defaulted
targetSnrMgn	Target Signal to Noise Ratio (SNR)	Yes	Yes	Yes
maxSnrMgn	Maximum SNR	Yes	Yes	Yes
minSnrMgn	Minimum SNR	Yes	Yes	Yes
downshiftSnrMgn	Seamless Rate Adaptation	no	NA	no
upshiftSnrMgn	Seamless Rate Adaptation	no	NA	no
minUpshiftTime	Seamless Rate Adaptation	no	NA	no
minDownshiftTime	Seamless Rate Adaptation	no	NA	no
fastMinTxRate	Minimum Transmit Rate for channels configured as Fast	Yes	Yes	Yes
interleaveMinTxRate	Minimum Transmit Rate for channels configured as Interleaved	Yes	Yes	Yes
fastMaxTxRate	Maximum Transmit Rate for channels configured as Fast	Yes	Yes	Yes
maxInterleaveDelay	Maximum Interleave Delay for channel(s) configured as Interleaved	Yes	1 to 63	63 when in ADSL2+ Annex A

Table 49: adsl-co-profile (Continued)

adsl-co-profile	ATU-C	SLMS Supported	SLMS Range supported	SLMS Default
interleaveMaxTxRate	Maximum Transmit Rate for channels configured as Interleaved	Yes	Yes	Yes
thresh15MinLofs	Loss of Frame event count	Yes	Yes	Yes
thresh15MinLoss	Loss of signal event count	Yes	Yes	Yes
thresh15MinLols	Loss of link event count	Yes	Yes	Yes
thresh15MinLprs	Loss of Loss of Power Seconds event count	Defaulted	Defaulted	Defaulted
thresh15MinESs	Errored Seconds event count	Yes	Yes	Yes
threshFastRateUp	Threshold time for increase rate on channels configured as Fast	Defaulted	Defaulted	Defaulted

Configuring S=1/2 transmission mode for fast mode

1 Verify that the **adminstatus** of the ADSL2+ port is *up*:

```
zSH> port up 1-17-1-0/adsl
1-17-1-0/adsl set to admin state UP
```

2 Verify that the ADSL2+ channelization is set to *fastonly*:

```
zSH> update adsl-profile 1/17/1
adsl-profile 1/17/1
Please provide the following: [q]uit.
adslLineConfProfile: -----> {0000000330}: ** read-only **
adslAlarmConfProfile: -----> {0000000330}: ** read-only **
adslTrellisModeEnabled: -----> {true}:
adslNTRModeEnabled: -----> {false}:
adslTransmissionMode: -----> {autonegotiatemode}:
adslChannelMode: -----> {interleaveonly}: fastonly
adslMaxDownstreamToneIndex: -----> {511}:
adslMinDownstreamToneIndex: -----> {32}:
adslMaxUpstreamToneIndex: -----> {31}:
adslMinUpstreamToneIndex: -----> {6}:
adslPotsBypassRelayMaxDuration: -> {60}:
adslLineDMTConfMode: -----> {freqdivmux}:
adslAnnexMModeEnabled: -----> {false}:
adslAnnexMPsdMask: -----> {eu56}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

- 3 Set the maximum transmit rate to 12 Mbps for fast ADSL2+ channel modes. This forces the ADSL2+ port into S=1/2 transmission mode.

```

zSH> update adsl-co-profile 1/17/1
adsl-co-profile 1/17/1
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {0}:
upshiftSnrMgn: -----> {90}:
minUpshiftTime: -----> {0}:
minDownshiftTime: -----> {0}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {32736000}: 12000000
maxInterleaveDelay: -----> {63}:
interleaveMaxTxRate: -----> {32736000}:
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLols: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ----> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
initFailureTrapEnable: ----> {disable}:
reachExtendedAdsl2: -----> {enable}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
txPowerAttenuation: -----> {20}:
cabMode: -----> {0}
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Configuring S=1/2 transmission mode for interleaved mode

- 1 Verify that the **adminstatus** of the ADSL2+ port is *up*:

```

zSH> port up 1-17-1-0/adsl
1-17-1-0/adsl set to admin state UP

```

- 2 Set the ADSL2+ channelization to *interleavedonly*:

```

zSH> update adsl-profile 1/17/1
adsl-profile 1/17/1

```

```

Please provide the following: [q]uit.
adslLineConfProfile: -----> {0000000330}: ** read-only **
adslAlarmConfProfile: -----> {0000000330}: ** read-only **
adslTrellisModeEnabled: -----> {true}:
adslNTRModeEnabled: -----> {false}:
adslTransmissionMode: -----> {autonegotiatemode}:
adslChannelMode: -----> {fastonly}: interleavedonly
adslMaxDownstreamToneIndex: -----> {511}:
adslMinDownstreamToneIndex: -----> {32}:
adslMaxUpstreamToneIndex: -----> {31}:
adslMinUpstreamToneIndex: -----> {6}:
adslPotsBypassRelayMaxDuration: -> {60}:
adslLineDMTConfMode: -----> {freqdivmux}:
adslAnnexMModeEnabled: -----> {false}:
adslAnnexMPsdMask: -----> {eu56}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

- 3 Set the maximum transmit rate to 12 Mbps for interleaved ADSL2+ channel mode. This forces the ADSL2+ port into S=1/2 transmission mode.

```

zSH> update adsl-co-profile 1/17/1
adsl-co-profile 1/17/1
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {0}:
upshiftSnrMgn: -----> {90}:
minUpshiftTime: -----> {0}:
minDownshiftTime: -----> {0}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {32736000}:
maxInterleaveDelay: -----> {63}:
interleaveMaxTxRate: -----> {32736000}: 12000000
thresh15MinLofs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLols: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
initFailureTrapEnable: ----> {disable}:
reachExtendedAdsl2: -----> {enable}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}:
phyRSupport: -----> {disable}:

```



```

phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
txPowerAttenuation: -----> {20}:
cabMode: -----> {0}
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Configure Broadcom Phy-R™ parameters

Setting the Broadcom Phy-R™ parameters in the CO and CPE ADSL2+ profiles is for advanced users.



Note: The Phy-R™ parameter in the ADSL2+ CO profile cannot be used unless there is a Broadcom CPE modem at the customer site with Phy-R™ parameters in the ADSL2+ CPE profile.

[Table 50](#) describes the profiles and parameters and suggested values to enable Phy-R™.

Table 50: Profiles and parameters used to configure ADSL2+ for Phy-R™

Parameter	Definition
adsl-co-profile	maxInterleaveDelay: 4 minINP: 20 phyRSupport: enable
adsl-cpe-profile	maxInterleaveDelay: 4 minINP: 20 phyRSupport: enable

Enabling Phy-R™ parameters

Update the **adsl-co-profile** and the **adsl-cpe-profile**:

```

zSH> update adsl-co-profile 1/3/2
adsl-co-profile 1/3/2
Please provide the following: [q]uit.
rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {0}:
upshiftSnrMgn: -----> {0}:
minUpshiftTime: -----> {0}:
minDownshiftTime: -----> {0}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:

```

```

fastMaxTxRate: -----> {32736000}:
maxInterleaveDelay: -----> {8}4
interleaveMaxTxRate: -----> {32736000}:
thresh15MinLoFs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLols: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
initFailureTrapEnable: ----> {disable}:
reachExtendedAdsl2: -----> {enable}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}20
phyRSupport: -----> {disable}enable
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
txPowerAttenuation: ----> {20}:
cabMode: -----> {0}

```

.....
Save changes? [s]ave, [c]hange or [q]uit:
Record updated.

zSH> update adsl-cpe-profile 1/3/2

adsl-cpe-profile 1/3/2

Please provide the following: [q]uit.

```

rateMode: -----> {adaptatruntime}:
rateChanRatio: -----> {50}:
targetSnrMgn: -----> {60}:
maxSnrMgn: -----> {310}:
minSnrMgn: -----> {0}:
downshiftSnrMgn: -----> {30}:
upshiftSnrMgn: -----> {90}:
minUpshiftSnrMgn: -----> {60}:
minDownshiftSnrMgn: -----> {60}:
fastMinTxRate: -----> {32000}:
interleaveMinTxRate: -----> {32000}:
fastMaxTxRate: -----> {1024000}:
interleaveMaxTxRate: -----> {1536000}:
maxInterleaveDelay: -----> {8}4
thresh15MinLoFs: -----> {0}:
thresh15MinLoss: -----> {0}:
thresh15MinLprs: -----> {0}:
thresh15MinESs: -----> {0}:
threshFastRateUp: -----> {0}:
threshInterleaveRateUp: ---> {0}:
threshFastRateDown: -----> {0}:
threshInterleaveRateDown: -> {0}:
minTxThresholdRateAlarm: --> {0}:
minINP: -----> {0}20
phyRSupport: -----> {disable}enable

```

```

phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit:
Record updated.

```

ADSL2+ statistics

Verifying the interface

Use the **dslstat** command to display the statistics on a ADSL2+ interface:

```
zSH> dslstat 1-9-1-0/adsl
```

General Stats:

```

-----
AdminStatus.....UP
LineStatus.....DATA
Line uptime (DD:HH:MM:SS).....3:19:53:22
DslUpLineRate (bitsPerSec).....1023000
DslDownLineRate (bitsPerSec).....22254800
DslMaxAttainableUpLineRate (bitsPerSec).....1169200
DslMaxAttainableDownLineRate (bitsPerSec)....25560000
Out Octets.....115964
Out Pkts/Cells.....2188
Out Discards.....159
Out Errors.....159
In Octets.....5988788
In Pkts/Cells.....112996
In Discards.....0
In Errors.....0
ATM OCD Count.....0
ATM NCD Count.....0
ATM HEC Count.....3
ATM far-end OCD Count.....0
ATM far-end NCD Count.....0
ATM far-end HEC Count.....0

```

ADSL CPE Information:

```

-----
CPE Serial Number.....
CPE Vendor Id.....B5004244434D0000A2pB022g2
CPE Version Number.....A2pB022g2

```

ADSL Physical Stats:

```

-----
Actual Transmission connection standard.....ADSL2+
AdslAtucCurrLineSnrMgn (tenths dB).....95
AdslAtucCurrLineAtn (tenths dB).....0
AdslAtucCurrOutputPwr (tenths dB).....143
AdslAturCurrLineSnrMgn (tenths dB).....66

```

```

AdslAturCurrLineAtn (tenths dB).....20
AdslAturCurrOutputPwr (tenths dB).....8
LOFS.....0
LOLS.....0
LOSS.....0
ESS.....6
Inits.....1
Adsl connects.....1
Adsl disconnects.....0

near-end statistics:
-----
blocks received.....17340235
errored blocks received.....6
CRC errors on interleaved buffer.....0
CRC errors on fast buffer.....6
FEC corrected errors on interleaved buffer...0
FEC corrected errors on fast buffer.....0
background errored blocks received.....0
non-SES blocks received.....0
Severely Errored Seconds.....0
Unavailable Seconds.....0
Loss of Signal Seconds.....0
Seconds with one/more FECs.....0
Seconds declared as high BER.....0
Fast retrains.....0
Fast retrain failures.....0

far-end statistics:
-----
blocks received.....17972303
errored blocks received.....33551
CRC errors on interleaved buffer.....0
CRC errors on fast buffer.....11136
FEC corrected errors on interleaved buffer...0
FEC corrected errors on fast buffer.....22415
background errored blocks received.....0
non-SES blocks received.....0
Severely Errored Seconds.....24
Unavailable Seconds.....381834
Loss of Signal Seconds.....24
Seconds with one/more FECs.....323
Loss of Power (dying gasps).....0
Seconds declared as high BER.....22415

phyR Statistics:
-----
Atuc Retransmitted codewords.....0
Atuc Corrected Retransmitted codewords.....0
Atuc UnCorrectableRetransmitted codewords....0
Atur Retransmitted codewords.....0
Atur Corrected Retransmitted codewords.....0
Atur UnCorrectable Retransmitted codewords...0

```

Clearing DSL counters (ADSL2+)

You can clear DSL counters to make identifying the changing statistics easier to read.

- 1 Clear the statistics using the **dslstat clear** command

```
zSH> dslstat clear 1-9-1-0/adsl
```

- 2 View the changes

After entering the dslstat command, (see [Verifying the interface on page 555](#)) to show the statistics before clearing the statistics. Statistics which are cleared by the **dslstat clear** command are in **bold**.

```
zSH> dslstat 1-9-1-0/adsl
```

General Stats:

```
-----
AdminStatus.....UP
LineStatus.....DATA
Line uptime (DD:HH:MM:SS).....0:01:07:48
DslUpLineRate (bitsPerSec).....1023000
DslDownLineRate (bitsPerSec).....22675400
DslMaxAttainableUpLineRate (bitsPerSec).....1173400
DslMaxAttainableDownLineRate (bitsPerSec)....25656000
Out Octets.....1017125120
Out Pkts/Cells.....0
Out Discards.....0
Out Errors.....0
In Octets.....742
In Pkts/Cells.....14
In Discards.....0
In Errors.....0
ATM OCD Count.....0
ATM NCD Count.....0
ATM HEC Count.....0
ATM far-end OCD Count.....0
ATM far-end NCD Count.....0
ATM far-end HEC Count.....0
```

ADSL CPE Information:

```
-----
CPE Serial Number.....
CPE Vendor Id.....B5004244434D0000A2pB022g2
CPE Version Number.....A2pB022g2
```

ADSL Physical Stats:

```
-----
Actual Transmission connection standard.....ADSL2+
AdslAtucCurrLineSnrMgn (tenths dB).....90
AdslAtucCurrLineAtn (tenths dB).....0
AdslAtucCurrOutputPwr (tenths dB).....143
AdslAturCurrLineSnrMgn (tenths dB).....59
AdslAturCurrLineAtn (tenths dB).....20
AdslAturCurrOutputPwr (tenths dB).....8
```

LOFS.....	0
LOLS.....	0
LOSS.....	0
ESS.....	0
Inits.....	0
Adsl connects.....	0
Adsl disconnects.....	0
near-end statistics:	

blocks received.....	330
errored blocks received.....	0
CRC errors on interleaved buffer.....	0
CRC errors on fast buffer.....	0
FEC corrected errors on interleaved buffer...	0
FEC corrected errors on fast buffer.....	0
background errored blocks received.....	0
non-SES blocks received.....	0
Severely Errored Seconds.....	0
Unavailable Seconds.....	0
Loss of Signal Seconds.....	0
Seconds with one/more FECs.....	0
Seconds declared as high BER.....	0
Fast retrains.....	0
Fast retrain failures.....	0
far-end statistics:	

blocks received.....	366
errored blocks received.....	0
CRC errors on interleaved buffer.....	0
CRC errors on fast buffer.....	0
FEC corrected errors on interleaved buffer...	0
FEC corrected errors on fast buffer.....	0
background errored blocks received.....	0
non-SES blocks received.....	0
Severely Errored Seconds.....	0
Unavailable Seconds.....	0
Loss of Signal Seconds.....	0
Seconds with one/more FECs.....	0
Loss of Power (dying gasps).....	0
Seconds declared as high BER.....	0

Table51 defines the statistics displayed in the **dslstat** command.

Table 51: ADSL2+ statistics

General statistics	Description
AdminStatus	Administrative status of the port: Values: Up Interface is ready to pass packets. Down Interface is unable to pass packets. Testing Interface is in a special testing state and is unable to pass packets.
LineStatus	Provides information about the ADSL link. Values: DOWN — no connection to CPE DOWNLOADING — downloading s/w to CPE DATA — connection established, passing data TEST — loopback or some other test function modem handshake phases (TRAINING, HANDSHAKE, DISCOVERY, ANALYSIS) SELT — selt in progress DELT —delt in progress UNKNOWN — an unknown error occurred (bad s/w, incorrect s/v version. etc.)
Line uptime (DD:HH:MM:SS)	How long the interface has been up in dd hh mm (day, hour, minute, second) format.
DslUpLineRate (bitsPerSec)	Displays the DSL upstream (customer premise > central office) line rate on this interface.
DslDownLineRate (bitsPerSec)	Displays the DSL downstream (central office > customer premise) line rate on this interface.
DslMaxAttainableUpLineRate (bitsPerSec)	Displays the maximum line rate that can be supported on this line in the upstream direction.
DslMaxAttainableDownLineRate (bitsPerSec)	Displays the maximum line rate that can be supported on this line in the downstream direction.
Out Octets	Number of transmitted octets.
Out Pkts/Cells	Number of transmitted packets/cells
Out Discards	Number of transmission discards.
Out Errors	Number of transmission errors.
In Octets	Number of received octets.
In Pkts/Cells	Number of transmitted packets/cells
In Discards	Number of received discards.
In Errors	Number of receive errors.

Table 51: ADSL2+ statistics (Continued)

General statistics	Description
ATM OCD Count	The number Out of Cell Delineation (OCD) events. An Out of Cell Delineation event is defined as seven consecutive ATM cells with Header Error Control (HEC) violations. A high number of these events may indicate a problem with the ATM TC sublayer.
ATM NCD Count	The number of far end No Cell Delineation (NCD) events on the far end.
ATM HEC Count	Number of corrected HEC cells.
ATM far-end OCD Count	The number Out of Cell Delineation (OCD) events. An Out of Cell Delineation event is defined as seven consecutive ATM cells with Header Error Control (HEC) violations. A high number of these events may indicate a problem with the ATM TC sublayer.
ATM far-end NCD Count	The number of far end No Cell Delineation (NCD) events on the far end.
ATM far-end HEC Count	Number of corrected HEC cells at the far-end.
ADSL CPE Information:	
CPE Serial Number	The vendor's serial number for the ADSL CPE device. The displayed information depends on the information the remote modem supplies.
CPE Vendor Id	The vendor portion of the ADSL CPE devices MAC address. The displayed information depends on the information the remote modem supplies.
CPE Version Number	The version number of the software of the ADSL CPE device. The displayed information depends on the information the remote modem supplies.
ADSL Physical Stats:	
Actual Transmission connection standard	Displays the maximum line rate that can be supported on this line in the downstream direction. Values: GHS GDMT T1 GLite Full Rate AutoNegotiate
AdslAtucCurrLineSnrMgn (tenths dB)	SNR Margin is the maximum increase in dB of the noise power received at the ATU-C on upstream direction), such that the BER requirements are met for all bearer channels received at the ATU. It ranges from 640 to 630 units of 0.1 dB (Physical values are -64 to 63 dB).

Table 51: ADSL2+ statistics (Continued)

General statistics	Description
AdslAtucCurrLineAtn (tenths dB)	Measured difference in the total power transmitted by the peer ATU-C and the total power received by this ATU.
AdslAtucCurrOutputPwr (tenths dB)	Actual Aggregate Transmit Power from the ATU-C on upstream direction at the instant of measurement. It ranges from -310 to 310 units of 0.1 dB (Physical values are -31 to 31 dBm).
AdslAturCurrLineSnrMgn (tenths dB)	SNR Margin is the maximum increase in dB of the noise power received at the ATU (ATU-R on downstream direction, such that the BER requirements are met for all bearer channels received at the ATU. It ranges from 640 to 630 units of 0.1 dB (Physical values are -64 to 63 dB).
AdslAturCurrLineAtn (tenths dB)	Measured difference in the total power transmitted by the peer ATU-R and the total power received by this ATU.
AdslAturCurrOutputPwr (tenths dB)	Actual Aggregate Transmit Power from the ATU (ATU-R on downstream direction at the instant of measurement. It ranges from -310 to 310 units of 0.1 dB (Physical values are -31 to 31 dBm).
LOFS	Number of Loss of Frame Seconds.
LOLS	Number of Loss of Line Seconds.
LOSS	Number of Loss of Signal Seconds.
ESS	Number of errored seconds (the number of one-second intervals containing one or more CRC anomalies or one or more LoS or Sef defects) that has been reported in the current 15-minute interval.
Inits	Number of line initialization attempts, including both successful and failed attempts.
Adsl connects	Number of successful connects at the near end since the agent reset.
Adsl disconnects	Number of disconnects at the near end since the agent reset.
near-end statistics:	
blocks received	Number of received blocks at the near end. This statistic is not incremented while there is a UAS or SES error on the interface.
errored blocks received	Number of background errored blocks at the near end. A background block error is an errored block that does not occur as part of a SES. A block refers to Reed Solomon error correction blocks. They are typically 255 bytes of data, and may span several symbols of data, regardless of how many or what parts of ATM cells they represent. This statistic is not incremented while there is a UAS or SES error on the interface.

Table 51: ADSL2+ statistics (Continued)

General statistics	Description
CRC errors on interleaved buffer	Number of cyclic redundancy check (CRC) errors on interleaved buffer at the near end. This statistic is not incremented while there is a UAS or SES error on the interface.
CRC errors on fast buffer	Number of cyclic redundancy check (CRC) errors on fast buffer at the near end. This statistic is not incremented while there is a UAS or SES error on the interface.
FEC corrected errors on interleaved buffer	Number of forward error corrections (FECs) on interleaved buffer at the near end. Forward error correction (Reed Solomon) is applied to the transported data. This process obtains coding gain, resulting in the link requiring lower signal-to-noise ratios (SNRs) for a given data and error rate. This process allows an increase in the data rate under given loop conditions. In addition, interleaving can be applied on top of error correction to obtain a higher degree of protection against error bursts or temporary loss of the data signal. The interleave distributes the data errors over multiple symbols. This action is intended to reduce the number of errors per Reed Solomon codeword to a number within the correction capability of the code. This statistic is not incremented while there is a UAS or SES error on the interface.
FEC corrected errors on fast buffer	Number of forward error corrections (FECs) on fast buffer at the near end. This statistic is not incremented while there is a UAS or SES error on the interface. Fast Buffer—Each ADSL frame consists of two parts, one from each of two buffers: the fast buffer and the interleaved buffer. The fast buffer, in addition to user data, may contain CRC error checking bits, and forward error correcting bits. The fast byte in frame 1, 34, and 35 contain indicator bits used for administrative functions. The interleaved buffer contains purely data.
background errored blocks received	Background errored blocks at near end. A background block error is an errored block that does not occur as part of a SES. A block refers to Reed Solomon error correction blocks. They are typically 255 bytes of data, and may span several symbols of data, regardless of how many or what parts of ATM cells they represent. This statistic is not incremented while there is a UAS or SES error on the interface.
non-SES blocks received	Number of non severely errored seconds (SES) blocks received at the near end.

Table 51: ADSL2+ statistics (Continued)

General statistics	Description
Severely Errored Seconds	Number of severely errored seconds (SES) at the near end. This is the number of 1-second intervals with any of the following error conditions: 18 or more CRC-8 anomalies over all received channels). If a CRC occurs over multiple bearer channels, then each related CRC-8 anomaly is counted only once for the whole set of bearer channels over which the CRC is applied. one or more LOS defects one or more SEF defects one or more LPR defects
Unavailable Seconds	Number of unavailable seconds (UAS) at the near end. This is the number of 1-second intervals for which the ADSL line is unavailable. The ADSL line becomes unavailable after the onset of 10 consecutive severely errored seconds (SESs). Note that the 10 SESs are included in unavailable time. The ADSL line becomes available after 10 consecutive seconds with no SESs. Note that the 10 seconds with no SESs are excluded from unavailable time.
Loss of Signal Seconds	Retrieved via OAM. Count of 1-second intervals containing one or more near end loss of signal (LOS) defects. An LOS failure is declared for either of the following reasons: after 2.5 ± 0.5 seconds of continuous LOS defects if LOS defect is present when a LOF occurs. A line circuit reports a LOS defect when the received power has fallen below the threshold. The threshold is set at 6 dB below the reference power. A LOS failure is cleared after 10 ± 0.5 seconds of no LOS defects. The most common cause for this fault is that the customer premises equipment (CPE) has been turned off. Supported for ADSL2/ADSL2plus only. This statistic is not incremented while there is a UAS or SES error on the interface.
Seconds with one/more FECs	Number of seconds with one or more forward error corrections (FECs) at the near end. These blocks are passed on as good data. This statistic is not incremented while there is a UAS or SES error on the interface.
Seconds declared as high BER	Number of seconds with high Bit Error Rate (BER).
Fast retrains	Number of fast retrains.
Fast retrain failures	Number of fast retrain failures.
far-end statistics:	

Table 51: ADSL2+ statistics (Continued)

General statistics	Description
blocks received	Number of received blocks at the far end. This statistic is not incremented while there is a UAS or SES error on the interface.
errored blocks received	Number of background errored blocks at the far end. A background block error is an errored block that does not occur as part of a SES. A block refers to Reed Solomon error correction blocks. They are typically 255 bytes of data, and may span several symbols of data, regardless of how many or what parts of ATM cells they represent. This statistic is not incremented while there is a UAS or SES error on the interface.
CRC errors on interleaved buffer	Number of cyclic redundancy check (CRC) errors on interleaved buffer at the far end. This statistic is not incremented while there is a UAS or SES error on the interface.
CRC errors on fast buffer	Number of cyclic redundancy check (CRC) errors on fast buffer at the far end. This statistic is not incremented while there is a UAS or SES error on the interface.
FEC corrected errors on interleaved buffer	Number of forward error corrections (FECs) on interleaved buffer at the far end. Forward error correction (Reed Solomon) is applied to the transported data. This process obtains coding gain, resulting in the link requiring lower signal-to-noise ratios (SNRs) for a given data rate and error rate. This process allows an increase in the data rate under given loop conditions. In addition, interleaving can be applied on top of error correction to obtain a higher degree of protection against error bursts or temporary loss of the data signal. The interleave distributes the data errors over multiple symbols. This action is intended to reduce the number of errors per Reed Solomon codeword to a number within the correction capability of the code. This statistic is not incremented while there is a UAS or SES error on the interface.

Table 51: ADSL2+ statistics (Continued)

General statistics	Description
FEC corrected errors on fast buffer	Number of forward error corrections (FECs) on fast buffer at the far end. This statistic is not incremented while there is a UAS or SES error on the interface. Fast Buffer—Each ADSL frame consists of two parts, one from each of two buffers: the fast buffer and the interleaved buffer. The fast buffer, in addition to user data, may contain CRC error checking bits, and forward error correcting bits. The fast byte in frame 1, 34, and 35 contain indicator bits used for administrative functions. The interleaved buffer contains purely data.
background errored blocks received	Number of background errored blocks at the far end. A background block error is an errored block that does not occur as part of a SES. A block refers to Reed Solomon error correction blocks. They are typically 255 bytes of data, and may span several symbols of data, regardless of how many or what parts of ATM cells they represent. This statistic is not incremented while there is a UAS or SES error on the interface.
non-SES blocks received	Number of non severely errored seconds (SES) blocks received at the far end.
Severely Errored Seconds	Number of errored seconds (the number of one-second intervals containing one or more cyclic redundancy check [CRC] anomalies or one or more loss of signal [LOS] defects) that has been reported in the current 15-minute interval. This statistic is not incremented while there is a UAS or SES error on the interface.
Unavailable Seconds	Number of unavailable seconds (UAS) at the far end. This is the number of 1-second intervals for which the ADSL line is unavailable. The ADSL line becomes unavailable after the onset of 10 consecutive severely errored seconds (SEs). Note that the 10 SEs are included in unavailable time. The ADSL line becomes available after 10 consecutive seconds with no SEs. Note that the 10 seconds with no SEs are excluded from unavailable time.
Loss of Signal Seconds	Loss of signal seconds at the near end.
Seconds with one/more FECs	Number of seconds with one or more forward error corrections (FECs) at the far end. These blocks are passed on as good data. This statistic is not incremented while there is a UAS or SES error on the interface.
Loss of Power (dying gasps)	The ATU-R (remote) device sends a dying-gasp message before it goes down so that the ATU-C (central office) device can differentiate between line down and ATU-R device down events.

Table 51: ADSL2+ statistics (Continued)

General statistics	Description
Seconds declared as high BER	Seconds declared as high BER by the far end.
phyR Statistics:	
Atuc Retransmitted codewords	ATUC Retransmitted Codewords.
Atuc Corrected Retransmitted codewords	ATUC Retransmitted corrected Codewords.
Atuc UnCorrectableRetransmitted codewords	ATUC Retransmitted uncorrectable Codewords.
Atur Retransmitted codewords	ATUR Retransmitted Codewords.
Atur Corrected Retransmitted codewords	ATUR Retransmitted corrected Codewords.
Atur UnCorrectable Retransmitted codewords	ATUR Retransmitted uncorrectable Codewords.

ADSL2+ bonding

The MXK ADSL2+ line cards support ADSL2+ bonding using the **bond add group** and the **bond add member** commands.

Bonding allows multiple lines to work together as a single line. Each bonding group can have:

- Two members per bond group.
- Members of a bond group must be contiguous ports which do not cross chip core boundaries. Each chip core has six ports (ports 1-6, 7-12, 13-18, 19-24, and so on). You can add port 1 and 2, 2 and 3, 3 and 4, 4 and 5, 5 and 6, but you cannot combine ports 6 and 7 because that would cross a chip core boundary.
- Bond group numbers must be in appropriate ranges. When using CLI to create a gbond group, the valid range for a group is from 1-48. Using ZMS to create a gbond group the valid range is from 1-48.

If you attempt to add more than two members, non-contiguous ports, ports which cross chip boundaries, or groups outside of the valid range the CLI will remind you of these rules. You also cannot add the same member to different bond groups.

Creating a gbond group on an ADSL2+ card

Create a single *gbond* group by first creating the bond group, then adding the members of the *gbond* group.

- 1 Create the *gbond* group with the **bond add group** command:

```
zSH> bond add group 1-9-1-0/gbond
```

2 Add members to the *gbond* group with the **bond add member** command:

```
zSH> bond add member 1-9-1-0/gbond 1-9-1-0/adsl
```

```
zSH> bond add member 1-9-1-0/gbond 1-9-2-0/adsl
```

3 View the bond group and the bond group members with the **bond show group** command:

```
zSH> bond show group 1-9-1-0/gbond
```

```

Bond Groups
Slot  GrpId  Type      State      Name
  9      1   gbond     OOS        1-9-1-0

Group Members
Slot  Port  Type      State      Name
  9    1   adsl      OOS        1-9-1-0
  9    2   adsl      OOS        1-9-2-0

```

View which *gbond* groups exist on a particular slot with the **bond group slot** command:

```
zSH> bond show slot 9
```

```

Bond Groups
Slot  GrpId  Type      State      Name
  9      1   gbond     OOS        1-9-1-0

```

The *gbond* group is displayed but does not show the bond group members.

Deleting a member of a gbond group

Use the **bond delete member** command to delete a member of a *gbond* group:

```
zSH> bond delete member 1-9-1-0/gbond 1-9-1-0/adsl
```

Deleting a gbond group

Use the **bond delete group** command to delete a *gbond* group:



Note: All *members* of the bond group must be deleted before the bond group can be deleted.

```
zSH> bond delete group 1-9-1-0/gbond
```

Moving members of a gbond group

After creating two *gbond* groups, a member can be moved from one bond group to another.

1 Create a *gbond* group:

```
zSH> bond add group 1-9-10-0/gbond
```

2 Add members to the *gbond* group:

```
zSH> bond add member 1-9-10-0/gbond 1-9-1-0/adsl
```

```
zSH> bond add member 1-9-10-0/gbond 1-9-2-0/adsl
```

3 View the *gbond* group and the members of the *gbond* group:

```
zSH> bond show group 1-9-10-0/gbond
```

```

Bond Groups
Slot  GrpId  Type      State      Name
  9      10   gbond     ACT        1-9-10-0
Group Members
Slot  Port   Type      State      Name
  9     1   adsl     ACT        1-9-1-0
  9     2   adsl     ACT        1-9-2-0

```

4 Create the next *gbond* group:

```
zSH> bond add group 1-9-11-0/gbond
```

5 Add a member to the *gbond* group:

```
zSH> bond add member 1-9-11-0/gbond 1-9-3-0/adsl
```

6 View existing *gbond* groups:

```
zSH> bond show all
```

```

Bond Groups
Slot  GrpId  Type      State      Name
  9      11   gbond     ACT        1-9-11-0
  9      10   gbond     ACT        1-9-10-0

```

View the *gbond* groups and their members:

```
zSH> bond show group 1-9-10-0/gbond
```

```

Bond Groups
Slot  GrpId  Type      State      Name
  9      10   gbond     ACT        1-9-10-0
Group Members
Slot  Port   Type      State      Name
  9     1   adsl     ACT        1-9-1-0
  9     2   adsl     ACT        1-9-2-0

```

```
zSH> bond show group 1-9-11-0/gbond
```

```

Bond Groups
Slot  GrpId  Type      State      Name
  9      11   gbond     ACT        1-9-11-0
Group Members
Slot  Port   Type      State      Name
  9     3   adsl     ACT        1-9-3-0

```


7 Move a member from *gbond* group *1-9-10-0/gbond* to *1-9-11-0/gbond*:

```
zSH> bond move 1-9-10-0/gbond 1-9-11-0/gbond 1-9-2-0/adsl
```

8 View the members in *gbond* group *1-9-10-0/gbond* and *1-9-11-0/gbond*:

```
zSH> bond show group 1-9-10-0/gbond
```

```

Bond Groups
  Slot  GrpId  Type      State      Name
   9     10   gbond     ACT        1-9-10-0
Group Members
  Slot  Port   Type      State      Name
   9     1    adsl      ACT        1-9-1-0

```

```
zSH> bond show group 1-9-11-0/gbond
```

```

Bond Groups
  Slot  GrpId  Type      State      Name
   9     11   gbond     ACT        1-9-11-0
Group Members
  Slot  Port   Type      State      Name
   9     3    adsl      ACT        1-9-3-0
   9     2    adsl      ACT        1-9-2-0

```

Member *1-9-2-0/adsl* moved from group *1-9-10-0/gbond* to *1-9-11-0/gbond*.

ATM on the ADSL2+ POTS line card

ATM data

The MXK MXK-ADSL2+-POTS-BCM-48A-2S line card communicates with subscriber integrated access devices (IADs) or DSL modems using ATM over DSL interfaces. The card relays the traffic to the uplink port, which provides a high-speed interface to an Ethernet network.

VPI and VCI ranges

[Table 52](#) lists the VPI/VCI support for the MXK. Note that VPI/VCI ranges can be changed.

Table 52: MXK VPI/VCI ranges

Interface	Default ranges
ADSL (per port)	VPI: 0 to 15 VCI: 32 to 255

Service categories

The MXK supports the following ATM service categories:

- Constant Bit Rate (CBR)
- non-real-time variable bit rate (nrt-VBR)
- real-time variable bit rate (rt-VBR)
- unspecified bit rate (UBR)

Constant Bit Rate (CBR)

The CBR service class is designed for ATM virtual circuits (VCs) needing a static amount of bandwidth that is continuously available for the duration of the active connection. An ATM VC configured as CBR can send cells at peak cell rate (PCR) at any time and for any duration. It can also send cells at a rate less than the PCR or even emit no cells.

Non-real-time variable bit rate (nrt-VBR)

The nrt-VBR service category is used by applications that are tolerant of network delays and do not require a timing relationship on each side of the connection. The nrt-VBR service supports somewhat bursty connections having less-stringent delay requirements than rt-VBR, but still require low cell loss. The source traffic descriptor is characterized by peak cell rate (PCR). nrt-VBR and UBR have the same priority in the MXK.

Real-time variable bit rate (rt-VBR)

The rt-VBR service category is used by applications that require a tightly constrained delay and delay variation. The source traffic descriptor is characterized by peak cell rate (PCR). rt-VBR has the highest priority in the MXK.

Unspecified bit rate (UBR)

The UBR service category does not specify traffic-related guarantees. No numerical commitments are made with respect to the cell loss ratio (CLR) experienced by the connection, or the cell transfer delay (CTD) experienced by the cells. With UBR service, the available bandwidth is fairly distributed to the active UBR subscribers. nrt-VBR and UBR have the same priority in the MXK.

Traffic descriptors

Each ATM endpoint requires a traffic descriptor, which defines the traffic parameters and type of service provided on ATM interfaces. Traffic descriptors are configured in **atm-traf-descr** records.



Note: ATM traffic policing and shaping are only supported in the downstream direction.

Quality of Service (QoS) parameters such as max cell transfer delay (maxCTD) and cell loss ratio (CLR) do not apply to a single node on the network and so are not provisioned for individual VCs.

Traffic descriptor parameters

Table 53 shows the required parameters used to define MXK traffic descriptors.

Table 53: ATM traffic descriptor parameters

TD type	td_param1	td_param2	td_param3	td_param4	td_param5
atmNoClpNoScr	PCR for CLP=0+1 traffic must be > 0	Not used	Not used	Not used	Not used

Tip: Refer to the following specifications for more information about traffic descriptors:

- ATM Forum, ATM User-Network Interface, Version 3.0 (UNI 3.0) Specification, 1994.
- ATM Forum, ATM User-Network Interface, Version 3.1 (UNI 3.1) Specification, November 1994.

ATM sample configurations

This section provides two ATM sample configurations, one for data and one for video applications.

ATM traffic descriptor example for data

Example for 1.5 Mbps downstream rate for data application.

```

zSH> new atm-traf-descr 1
atm-traf-descr 1
Please provide the following: [q]uit.
td_type: -----> {atmNoClpNoScr}:
td_param1: -----> {0}: 3661
td_param2: -----> {0}:
td_param3: -----> {0}:
td_param4: -----> {0}:
td_param5: -----> {0}:
cac-divider: -----> {1}:
td_service_category: -----> {ubr}: ubr
usage-parameter-control: -> {true}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s

```

New record saved.

ATM traffic descriptor example for video

Example for approximately 16 Mbps stream per port for video application.

```
zSH> new atm-traf-descr 1
atm-traf-descr 1
Please provide the following: [q]uit.
td_type: -----> {atmNoClpNoScr}:
td_param1: -----> {0}: 34736
td_param2: -----> {0}:
td_param3: -----> {0}:
td_param4: -----> {0}:
td_param5: -----> {0}:
cac-divider: -----> {1}:
td_service_category: -----> {ubr}: ubr
usage-parameter-control: -> {true}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

ATM statistics

Real-time ATM statistics on the MXK are provided through the NetHorizon ZMS client.

The ZMS performance manager periodically collects real-time statistical data. You can monitor real-time data at a polling interval of your choice. For information on how to access ZMS ATM statistics, refer to *the NetHorizon User's Guide* and the NetHorizon online help.

POTS to Voice over IP (VoIP) configuration

This section describe how to configure POTS to VoIP subscriber voice connections as follows:

- [VoIP overview, page572](#)
- [Basic VoIP configuration, page573](#)
- [Advanced VoIP configuration, page587](#)

VoIP overview

POTS subscribers can be connected to VoIP remote endpoints. For VoIP voice connections, several optional arguments such as *codec* are supported in the remote information of the **voice add** command. Supported codecs are:

- g711mu (the default setting)
- g711a

- g729a

The MXK G.729A VoIP compression provides an optional fallback mode to G.711. The parameter for the fallback mode is **g711-fallback** and is set in the **subscriber-voice-voip** profile. The default settings for the **subscriber-voice-voip** profile are:

- preferred-codec: g711mu
- g711-fallback: true (relevant with g729a)
- frames-per-packet: 4
- g726-byte-order: bigendian
- voip-password: password

The following VoIP remote information is available:

```
voip IpIfname dn dir-num [name username] [pw password] [plar dest-ipaddr]
[reg serverId] [codec pref-codec]
```



Note: For MGCP and Megaco calls, the MXK ignores the preferred-codec setting and selects the codec from a list provided by the MGCP server or media gateway controller.

Before creating VoIP connections, ensure that the IP interface for voice and VoIP server settings are properly configured.

Basic VoIP configuration

This section describes how to configure the VoIP signaling protocols supported by the MXK:

- [Configure a IP interface for voice traffic, page573](#)
- [SIP server configuration, page574](#)
- [SIP dial plan configuration, page576](#)
- [SIP PLAR server configuration, page578](#)
- [MGCP configuration, page579](#)
- [MEGACO \(H.248 configuration\), page581](#)
- [Additional VoIP features, page584](#)

By default, the MXK uses SIP signaling.

Configure a IP interface for voice traffic

Configure a network facing IP interface and route for voice traffic when configuring the MXK for any of the voice signalling protocols.

Configuring the IP voice path

Create a network facing IP interface that will pass the voice traffic to the network.

- 1 Create an IP interface, in this case on the network facing Ethernet port, and designate a VLAN, CoS and ToS values.

```
zSH> interface add 1-a-2-0/eth vlan 100 192.168.22.1/24
Created ip-interface-record ethernet2-100/ip.
```

Verify the IP interface.

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address          Media/Dest Address  IfName
-----
1/a/1/0/ip     UP      1 172.16.160.49/24  00:01:47:1a:fe:64   ethernet1
1/a/2/0/ip     DOWN    1 192.168.22.1/24   00:01:47:1a:fe:66   ethernet2-100
-----
```

- 2 Add a static route for the IP interface.

```
zSH> route add 10.10.10.0 255.255.255.0 192.168.22.254 1
```

Or add a default route.

```
zSH> route add default 192.168.22.254 1
```

View the route.

```
zSH> route list
Domain  Dest          Mask          Nexthop          IfNum  Cost  Enable
-----
1       0.0.0.0       0.0.0.0       172.16.160.254   0      1 enabled
1       10.10.10.0    255.255.255.0 192.168.22.254   0      1 enabled
```

SIP server configuration



Note: Redundant SIP server support is implemented through DNS lookups for only BroadSoft software configurations.

SIP signalling identifies callers and callees by SIP addresses and allows signals to be redirected to proxy servers.

The MXK supports single softswitch configurations.



Note: If all SIP calls do not register after a system reboot, increase the **server-max-timer** value in the **voice-system** profile to a higher value, for example 180 seconds. The default value is 20 seconds.



Note: Communication with SIP phones is only possible over one interface of the MXK. Communication with SIP phones that are in the same network as other interfaces on the MXK is not supported.

Configuring a SIP server

To configure SIP:

- 1 Create the **voip-server-entry** profiles to specify the VoIP server groups and IDs.

Specify the **voip-server-entry** profile with server group and server ID numbers. This example configures a SIP server in server group 1 with server ID 1.

```
zSH> new voip-server-entry 1/1
voip-server-entry 1/1
Please provide the following: [q]uit.
zhoneVoipServerAddrType: -----> {ipv4}:
zhoneVoipServerAddr: -----> {}: 192.168.49.1
zhoneVoipServerUdpPortNumber: -----> {5060}:
zhoneVoipServerId: -----> {generic}: metaswitch
protocol: -----> {sip}:
sendCallProceedingTone: -----> {false}:
rtcpEnabled: -----> {false}:
rtcpPacketInterval: -----> {5000}:
interdigitTimeOut: -----> {10}:
ipTos: -----> {0}:
systemDomainName: -----> {}:
expires-invite-value: -----> {3600}:
expires-register-value: -----> {3600}:
expires-header-method: -----> {register}:
session-timer: -----> {off}:
session-expiration: -----> {180}:
session-min-session-expiration: ---> {180}:
session-caller-request-timer: -----> {no}:
session-callee-request-timer: -----> {no}:
session-caller-specify-refresher: -> {omit}:
session-callee-specify-refresher: -> {uac}:
dtmf-mode: -----> {rfc2833}:
rtp-termid-syntax: -----> {}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

- 2 Create a SIP dialplan for the SIP server.

In each dialplan, specify the desired call parameters and use the **voip-server-entry** parameter to identify the server group for which the dialplan is used. This example references server group 1. See [SIP dial plan configuration on page 576](#) for more information.

```
zSH> new sip-dialplan 1
```

```

sip-dialplan 1
Please provide the following: [q]uit.
match-string: -----> {}: xT
sip-ip-address: -----> {0.0.0.0}: 192.168.49.1
destination-name: -----> {}:
number-of-digits: -----> {0}: 31
prefix-strip: -----> {}:
prefix-add: -----> {}:
dialplan-type: -----> {normal}:
voip-server-entry-index: -----> {0}: 1
override-interdigit-timeout: -> {0}: 3
.....
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.

```

- 3 Use the **voice add** command to add the POTS to VoIP connection. This examples creates a connection with a directory number 510-522-0401 and the name **smith**. The VoIP endpoint user name is case sensitive and must match the voice switch requirements, for example AAL/1 for MGCP with the Tekelec T6000 or TP/0001 for Megaco with Nortel CS2K.



Note: For MGCP and Megaco calls, the MXK ignores the **preferred-codec** setting and selects the codec from a list provided by the MGCP server or media gateway controller.

```

zSH> voice add pots 1-3-1-0/voicefxs voip ethernet3-100/ip DN 5105220401
name smith
Created subscriber-voice 1/2/1
Created subscriber-voice-pots 1004
Created subscriber-voice-voip 1005

```

- 4 View the voice connection.

```

zSH> voice show
Subscriber end-point          Remote end-point          Voice Prof Id  STA
-----
1-3-1-0/voicefxs            ethernet3-100/ip DN 5105220401/2/1      ENA
Total number of voice connections : 1

```

Caution: Avoid changes or deletions to the **ip-interface-record** profile after creating a voice connection on that interface.

SIP dial plan configuration

A dialing plan for POTS-to-SIP outgoing calls consists of a series of acceptable dial strings and the corresponding IP addresses to which SIP control messages are sent to initiate the call.

Each dial string is represented as digits, wildcards, and regular-expression-like patterns according to the following rules:

- Digits 0 to 9 are allowed as well as * and #.

- A wildcard ? represents any digit 0 to 9
- The character x to indicate a wildcard for 0 or more digits between 0-9.
- A dial-string character **T** can be used in the **override-interdigit-timeout** parameter value in the SIP dialplan.

Examples:

–**0T** for the number zero and nothing else.

–**011T** for numbers 011 then any number of digits before the interdigit time out.

–**9T** for the number 9 and any number of digits before the interdigit time out.

–**#T** anything followed by a # and an interdigit time out.

- A digit range can be specified using brackets [], as follows:
[135] means digits 1, 3, or 5.
[1-4] means digits 1, 2, 3, or 4.
- MGCP-style digit mapping where a period ‘.’ represents any digit and a | character indicates an inclusive OR.

Examples:

–**.T** for any number of digits before the interdigit timeout.

–***x.T** | **x.T** indicates star plus any number of digits followed by the inter-digit timeout or any number of digits followed by the inter-digit timeout.

– ***.xT** | **x.T** | **[2-9]11** indicates star plus any number of digits followed by the inter-digit timeout or any number of digits followed by the inter-digit timeout. or digits 2 to 9 followed by 11. The [2-9]11 explicit digit matching enables expedited call connections for emergency calls.

[Table54](#) describes the configurable **sip-dialplan** profile parameters for outgoing VoIP calls.

Table 54: sip-dialplan profile parameters

Parameter	Description
match-string	A dial string against which collected digits are matched.
sip-ip-address	Upon detecting a match between the collected digits and the dial string, this IP address is used for SIP negotiations to initiate the call.
destination -name	User-specified name of the destination for the dial string.
number-of-digits	Number of digits to wait for before initiating the call.
prefix-strip	Number of prefix digits to strip from dialled digits.
prefix-add	String to be added to the beginning of the dialled digits before call initiation.

Table 54: sip-dialplan profile parameters

Parameter	Description
dialplan-type	Type of the dial plan. Dialplan types are: • Normal • Call Park
voip-server-entry-index	An index to associated voip-server-entry for this sip-dialplan . This index references the registration server specified in the voip-server-entry profile.
override-interdigit-timeout	Override the partial-dial-timeout value in voice-system profile.

```

zSH> new sip-dialplan 1
Please provide the following: [q]uit.
match-string: -----> {}: 510555101[1-9]
sip-ip-address: -----> {0.0.0.0}: 192.16.88.199
destination-name: -----> {}: caller
number-of-digits: -----> {0}: 10
prefix-strip: -----> {0}: 1
prefix-add: -----> {}: 0
dialplan-type: -----> {normal}:
voip-server-entry-index: -----> {0}: 1
override-interdigit-timeout:--> {0}: 22
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.

```

SIP PLAR server configuration

SIP PLAR voice connections require the entry of the profile **voip-server-entry 255/255**.

Configuring a SIP PLAR server

This entry serves as the default server entry. The **zhoneVoipServerAddr** must be 0.0.0.0.

```

zSH> new voip-server-entry 255/255
voip-server-entry 255/255
Please provide the following: [q]uit.
zhoneVoipServerAddrType: -----> {ipv4}:
zhoneVoipServerAddr: -----> {}: 0.0.0.0
zhoneVoipServerUdpPortNumber: -----> {5060}:
zhoneVoipServerId: -----> {generic}:
protocol: -----> {sip}:
sendCallProceedingTone: -----> {false}:
rtcpEnabled: -----> {false}:
rtcpPacketInterval: -----> {5000}:
interdigitTimeOut: -----> {10}:
ipTos: -----> {0}:
systemDomainName: -----> {}:
expires-invite-value: -----> {3600}:

```

```

expires-register-value: -----> {3600}:
expires-header-method: -----> {register}:
session-timer: -----> {off}:
session-expiration: -----> {180}:
session-min-session-expiration: ---> {180}:
session-caller-request-timer: -----> {no}:
session-callee-request-timer: -----> {no}:
session-caller-specify-refresher: -> {omit}:
session-callee-specify-refresher: -> {uac}:
dtmf-mode: -----> {rfc2833}:
rtp-termid-syntax: -----> {}:
.....
Save new record? [s]ave, [c]hange or [q]uit:  s
New record saved.

```

MGCP configuration

MGCP signalling establishes call control elements or call agents to handle call control. MGCP devices execute the commands sent by the call agents. The MXK supports the voice message waiting indicator (VMWI) for MGCP connections.

The MXK supports two MGCP servers per VoIP system. In order to support multiple MGCP servers, the servers must be configured as redundant MGCP servers with redundant peer support enabled.

During the MXK system boot up, the MXK determines which redundant MGCP server to use. Then, during operations the MXK sends data to both the primary and the standby MGCP servers so that both MGCP servers are properly configured should a switch-over occur.

Configuring MGCP

To support multiple MGCP servers, create a **voip-server-entry** profile with a server group and server ID for each MGCP server. The first number in the `ifIndex` is for server group id and the second number is for the server ID. For example, 1/2 means server group 1 and server ID 2. The **voip-server-entry** profiles must use the same server group.



Note: Redundant MGCP softswitch configuration for Metaswitch ESA is configured by creating **voip-server-entry** profiles for each softswitch

This example creates voip-server-entry profiles for two MGCP servers using server group 1 and server IDs 1 and 2.



Note: The MGCP max call limiter is set at 288 calls. When the maximum number of allowable active calls is reached, the outgoing caller hears a congestion tone. For the incoming call, the phone does not ring.

To change the setting to MGCP:

1 Create the **voip-server-entry** profiles to enable MGCP:

```

zSH> new voip-server-entry 1/1
voip-server-entry 1/1
Please provide the following: [q]uit.
zhoneVoipServerAddrType: -----> {ipv4}:
zhoneVoipServerAddr: -----> {}: 172.16.160.1
zhoneVoipServerUdpPortNumber: -----> {5060}: 2727
zhoneVoipServerId: -----> {generic}: metaswitch
protocol: -----> {sip}: mgcp
sendCallProceedingTone: -----> {false}:
rtcpEnabled: -----> {false}:
rtcpPacketInterval: -----> {5000}:
interdigitTimeout: -----> {10}:
ipTos: -----> {0}:
systemDomainName: -----> {}:
expires-invite-value: -----> {3600}:
expires-register-value: -----> {3600}:
expires-header-method: -----> {register}:
session-timer: -----> {off}:
session-expiration: -----> {180}:
session-min-session-expiration: ---> {180}:
session-caller-request-timer: -----> {no}:
session-callee-request-timer: -----> {no}:
session-caller-specify-refresher: -> {omit}:
session-callee-specify-refresher: -> {uac}:
dtmf-mode: -----> {rfc2833}:
rtp-termid-syntax: -----> {}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.

```

```

zSH> new voip-server-entry 1/2
voip-server-entry 1/1
Please provide the following: [q]uit.
zhoneVoipServerAddrType: -----> {ipv4}:
zhoneVoipServerAddr: -----> {}: 172.16.160.3
zhoneVoipServerUdpPortNumber: -----> {5060}: 2727
zhoneVoipServerId: -----> {generic}: metaswitch
protocol: -----> {sip}: mgcp
sendCallProceedingTone: -----> {false}:
rtcpEnabled: -----> {false}:
rtcpPacketInterval: -----> {5000}:
interdigitTimeout: -----> {10}:
ipTos: -----> {0}:
systemDomainName: -----> {}:
expires-invite-value: -----> {3600}:
expires-register-value: -----> {3600}:
expires-header-method: -----> {register}:
session-timer: -----> {off}:
session-expiration: -----> {180}:
session-min-session-expiration: ---> {180}:
session-caller-request-timer: -----> {no}:
session-callee-request-timer: -----> {no}:

```

```

session-caller-specify-refresher: -> {omit}:
session-callee-specify-refresher: -> {uac}:
dtmf-mode: -----> {rfc2833}:
rtp-termid-syntax: -----> {}:
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.

```

- 2 The system will automatically reboot if the voice protocol is changed. After the reboot, verify that the **voip-server-entry** profile is configured for MGCP:

MEGACO (H.248 configuration)

The MEGACO protocol is used between elements of a physically decomposed multimedia gateway. The distributed multimedia gateway sub-components create a general framework used for gateways, multipoint control units and interactive voice response units (IVRs). Redundant Megaco servers are supported.

The MXK supports two VoIP servers per VoIP system. In order to support multiple VoIP servers, the servers must be configured as redundant VoIP servers with redundant peer support enabled.

During the MXK system boot up, the MXK determines which redundant VoIP server to use. Then, during operations the MXK sends data to both the primary and the standby VoIP servers so that both servers are properly configured should a switch-over occur.

To support multiple VoIP servers, create a **voip-server-entry** profile with a server group and server ID for each server. The first number in the ifIndex is for server group ID and the second number is for the server ID. For example, 1/2 means server group 1 and server ID 2. The **voip-server-entry** profiles must use the same server group.

Configuring MEGACO (H.248)

This example creates **voip-server-entry** profiles for two VoIP servers using server group 1 and server IDs 1 and 2.

To change the setting to MEGACO:

- 1 Create the **voip-server-entry** profiles to enable Megaco:

```

zSH> new voip-server-entry 1/1
voip-server-entry 1/1
Please provide the following: [q]uit.
zhoneVoipServerAddrType: -----> {ipv4}:
zhoneVoipServerAddr: -----> {}: 172.16.160.1
zhoneVoipServerUdpPortNumber: -----> {5060}: 2944
zhoneVoipServerId: -----> {generic}:
protocol: -----> {sip}: megaco
sendCallProceedingTone: -----> {false}:
rtcpEnabled: -----> {false}:

```

```

rtcpPacketInterval: -----> {5000}:
interdigitTimeOut: -----> {10}:
ipTos: -----> {0}:
systemDomainName: -----> {}:
expires-invite-value: -----> {3600}:
expires-register-value: -----> {3600}:
expires-header-method: -----> {register}:
session-timer: -----> {off}:
session-expiration: -----> {180}:
session-min-session-expiration: ---> {180}:
session-caller-request-timer: -----> {no}:
session-callee-request-timer: -----> {no}:
session-caller-specify-refresher: -> {omit}:
session-callee-specify-refresher: -> {uac}:
dtmf-mode: -----> {rfc2833}:
rtp-termid-syntax: -----> {}:

```

.....

Save new record? [s]ave, [c]hange or [q]uit: **s**

Record created.

zSH> new voip-server-entry 1/2

voip-server-entry 1/1

Please provide the following: [q]uit.

```

zhoneVoipServerAddrType: -----> {ipv4}:
zhoneVoipServerAddr: -----> {}: 172.16.160.3
zhoneVoipServerUdpPortNumber: -----> {5060}: 2944
zhoneVoipServerId: -----> {generic}:
protocol: -----> {sip}: megaco
sendCallProceedingTone: -----> {false}:
rtcpEnabled: -----> {false}:
rtcpPacketInterval: -----> {5000}:
interdigitTimeOut: -----> {10}:
ipTos: -----> {0}:
systemDomainName: -----> {}:
expires-invite-value: -----> {3600}:
expires-register-value: -----> {3600}:
expires-header-method: -----> {register}:
session-timer: -----> {off}:
session-expiration: -----> {180}:
session-min-session-expiration: ---> {180}:
session-caller-request-timer: -----> {no}:
session-callee-request-timer: -----> {no}:
session-caller-specify-refresher: -> {omit}:
session-callee-specify-refresher: -> {uac}:
dtmf-mode: -----> {rfc2833}:
rtp-termid-syntax: -----> {}:

```

.....

Save new record? [s]ave, [c]hange or [q]uit: **s**

Record created.

- 2 Set the **keep-alive-timer** for VoIP servers in the **voice-system** profile. The **server-max-timer** specifies the period between ServiceChange request messages. The **keep-alive-timer** specifies how often the MXK expects keep alive messages from the Gateway Controller.

If the MXK does not receive a keep alive message from the Gateway Controller in this interval, it sends an empty NTFY message to the controller. This should cause the controller to send a response.

If the MXK still does not receive a response to the NTFY message in a period equal to 4 times the **keep-alive-timer**, it will send a ServiceChange message to the Gateway Controller at an interval equal to the **keep-alive-timer**.

```
zSH> update voice-system 0
voice-system 0
Please provide the following: [q]uit.
hookflash-min-timer: -----> {100}:
hookflash-max-timer: -----> {1550}:
partial-dial-timeout: -----> {16}:
critical-dial-timeout: -----> {4}:
busy-tone-timeout: -----> {30}:
dial-tone-timeout: -----> {16}:
msg-wait-tone-timeout: -----> {16}:
offhook-warn-tone-timeout: -> {0}:
ringing-timeout: -----> {180}:
ringback-timeout: -----> {180}:
reorder-tone-timeout: -----> {30}:
stutter-tone-timeout: -----> {16}:
server-max-timer: -----> {20}:
config-max1: -----> {5}:
config-max2: -----> {7}:
max1-enable: -----> {true}:
max2-enable: -----> {true}:
max-waiting-delay: -----> {600}:
disconnection-wait-timer: --> {15}:
disconnection-min-timer: ---> {15}:
disconnection-max-timer: ---> {600}:
max-retransmit-timer: -----> {4}:
init-retransmit-timer: -----> {200}:
keep-alive-timer: -----> {60}: 90
no-response-timer: -----> {30}:
call-wait-max-repeat: -----> {2}:
call-wait-delay: -----> {10}:
pulse-inter-digit-timer: ---> {240}:
min-make-pulse-width: -----> {15}:
max-make-pulse-width: -----> {55}:
min-break-pulse-width: -----> {35}:
max-break-pulse-width: -----> {75}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Additional VoIP features

This section describes the configurable VoIP features for VoIP-enabled services.

- [Setting VoIP features, page584](#)
- [Changing the hookflash timer values, page584](#)

Setting VoIP features

To configure VoIP features:

```
zSH> update subscriber-voice 1/2/1
Please provide the following: [q]uit.
voice-connection-type: -----> {voiptopots}:  **
read-only **
voice-endpoint1-addr-index: ---> {1}: ** read-only **
voice-endpoint2-addr-index: ---> {1001}:  **
read-only **
voice-connection-description: -> {}:
voice-admin-status: -----> {enabled}:
huntgroup: -----> {false}:  **
read-only **
features: -----> {hookflash+onhooksignaling}:  hookflash
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Changing the hookflash timer values

The hookflash timer values can be configured to a specified range between minimum and maximum values. If hookflash is enabled on a VoIP subscriber, a hookflash is considered only if the onhook time is between the minimum and maximum timer values. Any time less than the minimum time setting is ignored and any time more than the maximum time setting is considered to be onhook.

[Table55](#) describes the configurable parameters in the **voice-system 0** profile that change hookflash timer settings.

Table 55: hookflash timer parameter values	
Parameter	Description
hookFlashTimerMin	Specifies the minimum hookflash timer value in milliseconds. Values: 0 to 2147483647 Default: 100 milliseconds
hookFlashTimerMax	Specifies the maximum hookflash timer value in milliseconds. Values: 0 to 2147483647 Default: 1550 milliseconds

To change the hookflash timer values:

```

zSH> update voice-system 0
Please provide the following: [q]uit.
hookflash-max-timer: -> {1550}: 2000
hookflash-min-timer: -> {100}: 500
partial-dial-timeout: -----> {16}:
critical-dial-timeout: -----> {4}:
busy-tone-timeout: -----> {30}:
dial-tone-timeout: -----> {16}:
msg-wait-tone-timeout: -----> {16}:
offhook-warn-tone-timeout: -> {0}:
ringing-timeout: -----> {180}:
ringback-timeout: -----> {180}:
reorder-tone-timeout: -----> {30}:
stutter-tone-timeout: -----> {16}:
server-max-timer: -----> {20}:
config-max1: -----> {5}:
config-max2: -----> {7}:
max1-enable: -----> {true}:
max2-enable: -----> {true}:
max-waiting-delay: -----> {600}:
disconnection-wait-timer: --> {15}:
disconnection-min-timer: ---> {15}:
disconnection-max-timer: ---> {600}:
max-retransmit-timer: -----> {4}:
init-retransmit-timer: -----> {200}:
keep-alive-timer: -----> {60}:
no-response-timer: -----> {30}:
call-wait-max-repeat: -----> {2}:
call-wait-delay: -----> {10}:
pulse-inter-digit-timer: ---> {100}:
min-make-pulse-width: -----> {25}:
max-make-pulse-width: -----> {55}:
min-break-pulse-width: -----> {45}:
max-break-pulse-width: -----> {75}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Advanced VoIP configuration

This section covers:

- [ToS, CoS, and sCoS on IP interfaces, page587](#)
- [ToS fields in an IP header, page587](#)
- [CoS and Tos for all IP packets, page589](#)
- [ToS configuration to add priority to voice packets, page590](#)

ToS, CoS, and sCoS on IP interfaces

The MXK supports the marking and remarking of ToS values in IP packets and Class of Service (CoS) values in Ethernet VLAN headers as defined by IETF RFC1349 and IEEE 802.1p respectively. The configured ToS and CoS levels specify packet priority and queueing used to transport the packet through the Ethernet and IP networks. The MXK sets and transports the ToS/CoS values, while the switches and routers connected to the MXK perform the QoS priority and queueing services.

CoS is set in tagged or s-tagged (VLANs or SLANs) on the outgoing interface and ToS is set in untagged (no VLANs or sLANs) configurations also on the outgoing interface. Both CoS and ToS can be set in the same outgoing interface to ensure that important services like voice and video retain packet priority and queueing throughout the network. As packets travel through the network, routers will recognize ToS values and switches will recognize CoS values.

See [ToS, CoS, and sCoS on IP interfaces on page118](#) for more general IP QoS information.

When configuring the MXK for voice, a separate ToS value for VoIP signalling packets is set in the **voip-server-entry** profile.

[Table56](#) specifies the IP ToS settings used in the **voip-server-entry** profile based on IP Precedence bits.

ToS fields in an IP header

IP packets have a ToS byte in their headers that contains information about relative priority. The ToS byte is divided into two fields called IP Precedence and ToS. The IP Precedence field contains a 3-bit priority designation. Most normal traffic has an IP Precedence value of zero. Higher values in this field indicate that traffic is more important and that it requires special treatment. IP Precedence values greater than 5 are reserved for network functions.

The ToS field indicates the queueing priority value based on eight (0-7) levels of service. This field contains information about how the traffic should be forwarded. The MXK supports basic ToS marking without queue servicing options in the **ip-interface-record** profile. Packets marked based on a configurable profile to let the system know which bits use which queue.

[Table 56](#) specifies the IP ToS settings used in the **voip-server-entry** profile based on IP Precedence bits.



Note: When setting ToS for IP packets in the **ip-interface-record** profile, the values in the precedence bits column are used, when setting ToS for voice signalling packets in the **voip-server-entry** profile, the values in the ToS value column are used.

Table 56: IP ToS settings and IP Precedence bits

Precedence bits	ToS value
0 (Routine)	0
1 (Priority)	32
2 (Immediate)	64
3 (Flash)	96
4 (Flash override)	128
5 (CRITIC/ECP.)	160
6 (Internetwork control)	192
7 (Network control)	224

View the **ipTos** parameter in the **voice-server-entry** profile.

```

zSH> show voip-server-entry
zhoneVoipServerAddrType:-----> unknown ipv4 ipv6 ipv4z ipv6z dns
zhoneVoipServerAddr:-----> {255}
zhoneVoipServerUdpPortNumber:-----> {1 - 65535}
zhoneVoipServerId:-----> longboard asterisk sipexpressrouter met
aswitch sylantr broadsoft ubiquity generalbandwidth tekelec-t6000 generic
sonus siemens tekelec-t9000 lucent-telica nortel-cs2000 nuera lucent-ime
rge coppercom newcross cisco-bts cirpack-utp italtel-issw cisco-pgw micro
trol-msk10 nortel-dms10 verso-clarent-c5cm cedarpoint-safari huawei-softx300
0 nortel-cs1500 aqua-t7000 utstarcom-mswitch broadsoft-broadworks broadsof
t-m6 genband-g9 netcentrex genband-g6
protocol:-----> sip mgcp megaco ncs
sendCallProceedingTone:-----> true false
rtcpEnabled:-----> true false
rtcpPacketInterval:-----> {0 - 0}
interdigitTimeOut:-----> {0 - 0}
ipTos:-----> {0 - 255}
systemDomainName:-----> {260}
expires-invite-value:-----> {0 - 2147483647}
expires-register-value:-----> {0 - 2147483647}
expires-header-method:-----> invite+register
session-timer:-----> on off
session-expiration:-----> {90 - 2147483647}
session-min-session-expiration:---> {90 - 2147483647}
session-caller-request-timer:-----> yes no

```

```

session-callee-request-timer:----> yes no
session-caller-specify-refresher:-> uac uas omit
session-callee-specify-refresher:-> uac uas
dtmf-mode:-----> inband rfc2833
rtp-termid-syntax:-----> {96}

```

CoS and Tos for all IP packets

Create Cos and ToS for all IP packets on a network facing IP interface.

Configuring CoS and ToS on an IP interface

- 1 Create an IP interface, in this case on the network facing Ethernet port, and designate a VLAN, CoS and ToS values.

```

zSH> interface add 1-a-2-0/eth vlan 100 cos 5 tosOrig 5 192.168.22.1/24
Created ip-interface-record ethernet2-100/ip.

```

Verify the IP interface.

```

zSH> interface show
2 interfaces

```

Interface	Status	Rd/Address	Media/Dest Address	IfName
1/a/1/0/ip	UP	1 172.16.160.49/24	00:01:47:1a:fe:64	ethernet1
1/a/2/0/ip	DOWN	1 192.168.22.1/24	00:01:47:1a:fe:66	ethernet2-100

View the **ip-interface-record** profile to verify the VLAN, CoS, and ToS values.

```

zSH> get ip-interface-record ethernet2-100/ip
ip-interface-record ethernet2-100/ip
vpi: -----> {0}
vci: -----> {0}
rdindex: -----> {1}
dhcp: -----> {none}
addr: -----> {192.168.22.1}
netmask: -----> {255.255.255.0}
bcastaddr: -----> {192.168.22.255}
destaddr: -----> {0.0.0.0}
farendaddr: -----> {0.0.0.0}
mru: -----> {1500}
reasmmxsize: -----> {0}
ingressfiltername: -----> {}
egressfiltername: -----> {}
pointtopoint: -----> {no}
mcastenabled: -----> {yes}
ipfwdenabled: -----> {yes}
mcastfwdenabled: -----> {yes}
natenabled: -----> {no}
bcastenabled: -----> {yes}
ingressPacketRuleGroupIndex: -> {0}
egressPacketRuleGroupIndex: --> {0}

```

```

ipaddrdynamic: -----> {static}
dhcpserverenable: -----> {false}
subnetgroup: -----> {0}
unnumberedindex: -----> {0}
mcastcontrollist: -----> {}
vlanid: -----> {100}
maxVideoStreams: -----> {0}
tosOption: -----> {originate}
tosCOS: -----> {5}
vlanCOS: -----> {5}
s-tagTPID: -----> {0x8100}
s-tagId: -----> {0}
s-tagIdCOS: -----> {0}

```

2 Add a static route for the IP interface.

```
zSH> route add 10.10.10.0 255.255.255.0 192.168.22.254 1
```

Or add a default route.

```
zSH> route add default 192.168.22.254 1
```

View the route.

```
zSH> route list
```

Domain	Dest	Mask	Nexthop	IfNum	Cost	Enable
1	0.0.0.0	0.0.0.0	172.16.160.254	0	1	enabled
1	10.10.10.0	255.255.255.0	192.168.22.254	0	1	enabled

ToS configuration to add priority to voice packets

ToS for voice signalling packets is set in the **voip-server-entry** profile.

Configuring VoIP QoS

To add ToS to signalling packets, you must configure the **ipTos** parameter of the **voip-server-entry** profile.

1 View the existing **voip-server-entry** profiles if necessary.

```
zSH> list voip-server-entry
voip-server-entry 1/1
1 entry found.
```

2 Configure the **ipTos** parameter with the ToS value (see [Table56](#)) in the **voip-server-entry** profile to add the ToS value to the signalling voice packets.

```
zSH> update voip-server-entry 1/1
voip-server-entry 1/1
Please provide the following: [q]uit.
zhoneVoipServerAddrType: -----> {ipv4}:
zhoneVoipServerAddr: -----> {172.16.160.3}:
zhoneVoipServerUdpPortNumber: -----> {5060}:
```

```

zhoneVoipServerId: -----> {generic}:
protocol: -----> {sip}:  ** read-only **
sendCallProceedingTone: -----> {false}:
rtcpEnabled: -----> {false}:
rtcpPacketInterval: -----> {5000}:
interdigitTimeOut: -----> {10}:
ipTos: -----> {0}:  160
systemDomainName: -----> {}:
expires-invite-value: -----> {3600}:
expires-register-value: -----> {3600}:
expires-header-method: -----> {register}:
session-timer: -----> {off}:
session-expiration: -----> {180}:
session-min-session-expiration: ---> {180}:
session-caller-request-timer: -----> {no}:
session-callee-request-timer: -----> {no}:
session-caller-specify-refresher: -> {omit}:
session-callee-specify-refresher: -> {uac}:
dtmf-mode: -----> {rfc2833}:
rtp-termid-syntax: -----> {}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

3 View the ToS value.

```

zSH> get voip-server-entry 1/1
voip-server-entry 1/1
zhoneVoipServerAddrType: -----> {ipv4}
zhoneVoipServerAddr: -----> {172.16.160.3}
zhoneVoipServerUdpPortNumber: -----> {5060}
zhoneVoipServerId: -----> {generic}
protocol: -----> {sip}
sendCallProceedingTone: -----> {false}
rtcpEnabled: -----> {false}
rtcpPacketInterval: -----> {5000}
interdigitTimeOut: -----> {10}
ipTos: -----> {160}
systemDomainName: -----> {}
expires-invite-value: -----> {3600}
expires-register-value: -----> {3600}
expires-header-method: -----> {register}
session-timer: -----> {off}
session-expiration: -----> {180}
session-min-session-expiration: ---> {180}
session-caller-request-timer: -----> {no}
session-callee-request-timer: -----> {no}
session-caller-specify-refresher: -> {omit}
session-callee-specify-refresher: -> {uac}
dtmf-mode: -----> {rfc2833}
rtp-termid-syntax: -----> {}

```

Emergency Stand Alone (ESA) SIP support

This section describes ESA SIP support on the MXK:

- [Configuring VoIP ESA clusters, page593](#)
- [Configuring ESA for 911 calls, page595](#)
- [Verifying ESA, page595](#)

For VoIP SIP or SIP PLAR voice connections, the MXK provides emergency calling services during network or equipment failures that cause a loss of connection to the configured SIP server.

For VoIP SIP or SIP PLAR connections, the ESA feature enables numbers configured within ESA dialplans to communicate with any residences or businesses specified as the destination of the dialplans in an ESA cluster of MXK devices. Incoming calls from outside the ESA group and outgoing calls to numbers outside the SIP dialplan receive a fast-busy signal.

When ESA is activated, call features such as call waiting, are not supported.

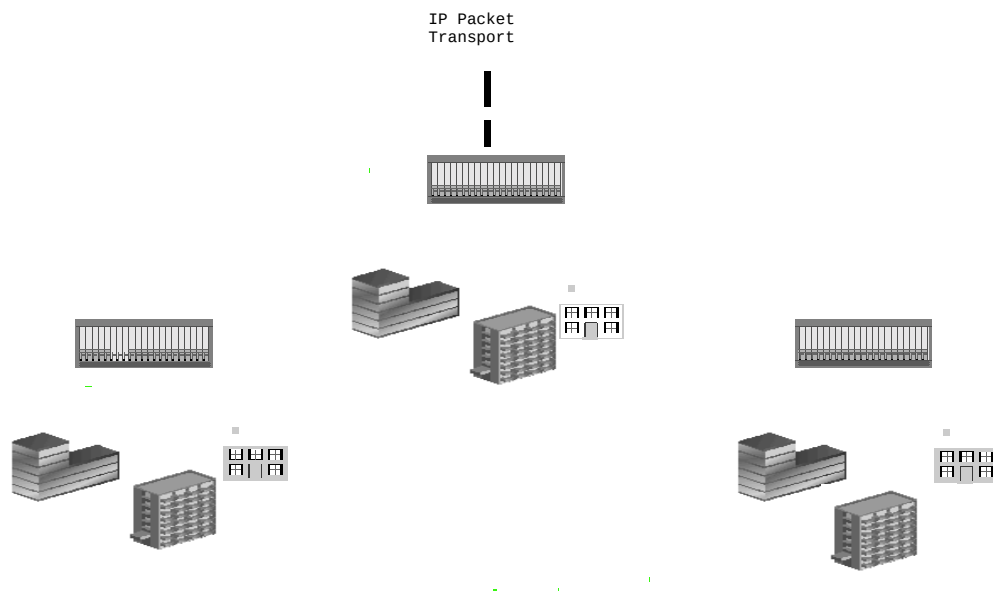


Note: After a loss of connection to the SIP server, there may be a delay up to 5 minutes before ESA notification is received and ESA features are accessible.

There maybe a similar delay before resuming normal calling after the outage is restored.

[Figure52](#) illustrates ESA support for VoIP SIP or SIP PLAR connections.

Figure 52: ESA for VoIP SIP or SIP PLAR connections



Configuring VoIP ESA clusters

To configure ESA clusters for VoIP connections, configure a VoIP server and create a dialplan for the VOIP server. Also, create an ESA dialplan for each of the MXK devices participating in the ESA cluster. For each ESA dialplan, enter the IP addresses of the desired MXK in the **sip-ip-address** field and change the **dialplan-type** to **esa**. Also, if desired, change the **destination-name** to the target MXK.

When in ESA mode, the MXK sequentially checks the configured dialplans for a matching string starting with the lowest number to the highest number dialplan. If a match is found, the call connection process is initiated immediately. If a match is not found, the next sequential dialplan is checked until all configured dialplans have been checked. Calls with unmatched strings are then terminated. It is recommended to configure lower number dialplans for more frequently called nodes and higher number dialplans for less frequently called nodes.

This example creates VoIP server 1/1 and creates SIP dialplan O for the VoIP server. SIP dialplan 1 is used on MXK 1 with IP address 172.24.94.219; SIP dialplan 2 is used on MXK 2 with IP address 172.24.94.222. SIP dialplan 3 is used on MXK 3 with IP address 172.24.94.223. It also sets the match-string to 'x' to accept all numbers, the number of digits to 7, and the dialplan type to ESA. This dialplan enables ESA calls to connect to other subscribers within the same MXK. Additional dialplans are created for each of the neighboring MXK nodes.



Note: A SIP dialplan of type normal should be configured and connected to a VoIP SIP server for non-ESA calls.

```
zSH> new voip-server-entry 1/1
Please provide the following: [q]uit.
zhoneVoipServerAddrType: -----> {ipv4}:
zhoneVoipServerAddr: -----> {}: 172.16.60.1
zhoneVoipServerUdpPortNumber: -----> {5060}:
zhoneVoipServerId: -----> {generic}:
protocol: -----> {sip}:
sendCallProceedingTone: -----> {false}:
rtcpEnabled: -----> {false}:
rtcpPacketInterval: -----> {5000}:
interdigitTimeout: -----> {10}:
ipTos: -----> {0}:
systemDomainName: -----> {}:
expires-invite-value: -----> {3600}
expires-register-value: -----> {3600}
expires-header-method: -----> {register}
session-expiration: -----> {0}
session-min-SE: -----> {-606348325}
session-caller-request-timer: -----> {no}
session-callee-request-timer: -----> {no}
session-caller-specify-refresher: -> {omit}
session-callee-specify-refresher: -> {uac}
```



```
omit-session-callee-specify-refresher:->(uac)
dtmf-mode:-----> (inband)
.....
Save new record? [s]ave, [c]hange or [q]uit: s
Record created.
```

```
zSH> new sip-dialplan 0
match-string: -----> {x}
sip-ip-address: -----> {0} 172.16.60.1
destination-name: -----> {} VoIP Server
number-of-digits: -----> {0} 7
prefix-strip: -----> {0}
prefix-add: -----> {}
dialplan-type: -----> {normal}
voip-server-entry-index: -----> {0} 1
override-interdigit-timeout: -> {0}
```

```
zSH> new sip-dialplan 1
match-string: -----> {x}
sip-ip-address: -----> {0} 172.24.94.219
destination-name: -----> {} MXK#1
number-of-digits: -----> {0} 7
prefix-strip: -----> {0}
prefix-add: -----> {}
dialplan-type: -----> {normal} esa
voip-server-entry-index: -----> {0}
override-interdigit-timeout: -> {0}
```

Create additional SIP dialplans for so ESA calls can connect to subscribers on other MXK devices. This dialplan allows ESA calls to connect to subscribers on MXK 2.

```
zSH> new sip-dialplan 2
match-string: -----> {x}
sip-ip-address: -----> {0} 172.24.94.222
destination-name: -----> {} MXK#2
number-of-digits: -----> {0} 7
prefix-strip: -----> {0}
prefix-add: -----> {}
dialplan-type: -----> {normal} esa
voip-server-entry-index: -----> {0}
override-interdigit-timeout: -> {0}
```

This dialplan allows ESA calls to connect to subscribers on MXK 3.

```
zSH> new sip-dialplan 3
match-string: -----> {x}
sip-ip-address: -----> {0} 172.24.94.223
destination-name: -----> {} MXK#3
number-of-digits: -----> {0} 7
prefix-strip: -----> {0}
prefix-add: -----> {}
dialplan-type: -----> {normal} esa
voip-server-entry-index: -----> {0}
```

```
override-interdigit-timeout: -> {0}
```

Configuring ESA for 911 calls

To configure ESA for VoIP connections for 911 calls, create an ESA dialplan with a match-string of 911 and the IP address of the MXK shelf in the **sip-ip-address** field. Also, change the number of digits and prefix-strip to 3. The prefix-strip setting deletes the dialed 911 numbers. Enter the desired phone number to be called in the prefix-add field. This number must be a valid voicefxs line in the same MXK shelf. Change the dial-plan type to **esa**.

This example creates a SIP dialplan called 911 on the MXK with IP address 172.24.94.219. It replaces the dialed 911 number with the phone number 7281001 and changes the dialplan type to ESA.

```
zSH> new sip-dialplan 911
match-string: -----> {} 911
sip-ip-address: -----> {0} 172.24.94.219
destination-name: -----> {}
number-of-digits: -----> {0} 3
prefix-strip: -----> {0} 3
prefix-add: -----> {} 7280004
dialplan-type: -----> {normal} esa
voip-server-entry-index: -----> {0}
override-interdigit-timeout: -> {0}
```

Verifying ESA

To verify whether ESA support is in-use, enter the **voice status** command. This command lists the voice port, destination, call state, and ESA state along with other status information.

```
zSH> voice status
```

port	term	state	destination	call state	hook	ring	ESA
----	-----	----	-----	-----	-----	-----	----
1-6-1-0/voicefxs		UP	VoIP:69:VoIP EndPtIdx-152	No call	ON	NoRing	ON
1-6-2-0/voicefxs		UP	VoIP:69:VoIP EndPtIdx-154	No call	ON	NoRing	ON
1-6-3-0/voicefxs		UP	GR303:IG-one:CRV-3	No call	ON	NoRing	N/A

ADSL2+ cable and port pinouts

This section describes the ADSL2+ cables available from Zhone Technologies and the ADSL2+ port pinouts.

- [ADSL-48 card pinouts, page596](#)
- [ADSL2+ cable pinouts, page600](#)

ADSL2+ card port pinouts

This section describes the following ADSL2+ port pinouts.

ADSL-48 card pinouts

[Table57](#) lists the ADSL-48 card pinouts.

Table 57: ADSL-48 card pinouts

Port	Signal	Pin
1TipJ7-2		
	Ring	J7-1
2TipJ7-4		
	Ring	J7-3
3TipJ7-6		
	Ring	J7-5
4TipJ7-8		
	Ring	J7-7
5TipJ7-10		
	Ring	J7-9
6TipJ7-12		
	Ring	J7-11
7TipJ7-14		
	Ring	J7-13
8TipJ7-16		
	Ring	J7-15
9TipJ7-18		
	Ring	J7-17
10	Tip	J7-20
	Ring	J7-19
11	Tip	J7-22
	Ring	J7-21
12	Tip	J7-24
	Ring	J7-23
13	Tip	J7-26
	Ring	J7-25

Table 57: ADSL-48 card pinouts (Continued)

Port	Signal	Pin
14	Tip	J7-28
	Ring	J7-27
15	Tip	J7-30
	Ring	J7-29
16	Tip	J7-32
	Ring	J7-31
17	Tip	J7-34
	Ring	J7-33
18	Tip	J7-36
	Ring	J7-35
19	Tip	J7-38
	Ring	J7-37
20	Tip	J7-40
	Ring	J7-39
21	Tip	J7-42
	Ring	J7-41
22	Tip	J7-44
	Ring	J7-43
23	Tip	J7-46
	Ring	J7-45
24	Tip	J7-48
	Ring	J7-47
25	Tip	J7-50
	Ring	J7-49
26	Tip	J7-52
	Ring	J7-51
27	Tip	J7-54
	Ring	J7-53
28	Tip	J7-56
	Ring	J7-55

Table 57: ADSL-48 card pinouts (Continued)

Port	Signal	Pin
29	Tip	J7-58
	Ring	J7-57
30	Tip	J7-60
	Ring	J7-59
31	Tip	J7-62
	Ring	J7-61
32	Tip	J7-64
	Ring	J7-63
33	Tip	J7-66
	Ring	J7-65
34	Tip	J7-68
	Ring	J7-67
35	Tip	J7-70
	Ring	J7-69
36	Tip	J7-72
	Ring	J7-71
37	Tip	J7-74
	Ring	J7-73
38	Tip	J7-76
	Ring	J7-75
39	Tip	J7-78
	Ring	J7-77
40	Tip	J7-80
	Ring	J7-79
41	Tip	J7-82
	Ring	J7-81
42	Tip	J7-84
	Ring	J7-83
43	Tip	J7-86
	Ring	J7-85

Table 57: ADSL-48 card pinouts (Continued)

Port	Signal	Pin
44	Tip	J7-88
	Ring	J7-87
45	Tip	J7-90
	Ring	J7-89
46	Tip	J7-92
	Ring	J7-91
47	Tip	J7-94
	Ring	J7-93
48	Tip	J7-96
	Ring	J7-95

ADSL2+ cable pinouts

This section provides the following information:

- [ADSL-48 to dual 50-pin connector cable, page600](#)
- [48-port ADSL to dual 50-pin connector cables, page604](#)
- [Variations of 48-port ADSL to dual 50-pin connector cables, page605](#)

ADSL-48 to dual 50-pin connector cable

[Figure53](#) shows the ADSL-48 to dual 50-pin connector cable (MALC-CBL-ADSL-48). [Table58 on page 601](#) lists the ADSL-48 card pinouts. [Table59 on page 605](#) lists additional ADSL-48 to dual 50-pin connector cables. [Table60 on page 605](#) lists variations of these cables.

Figure 53: 48-port ADSL2+ to dual 50-pin cable

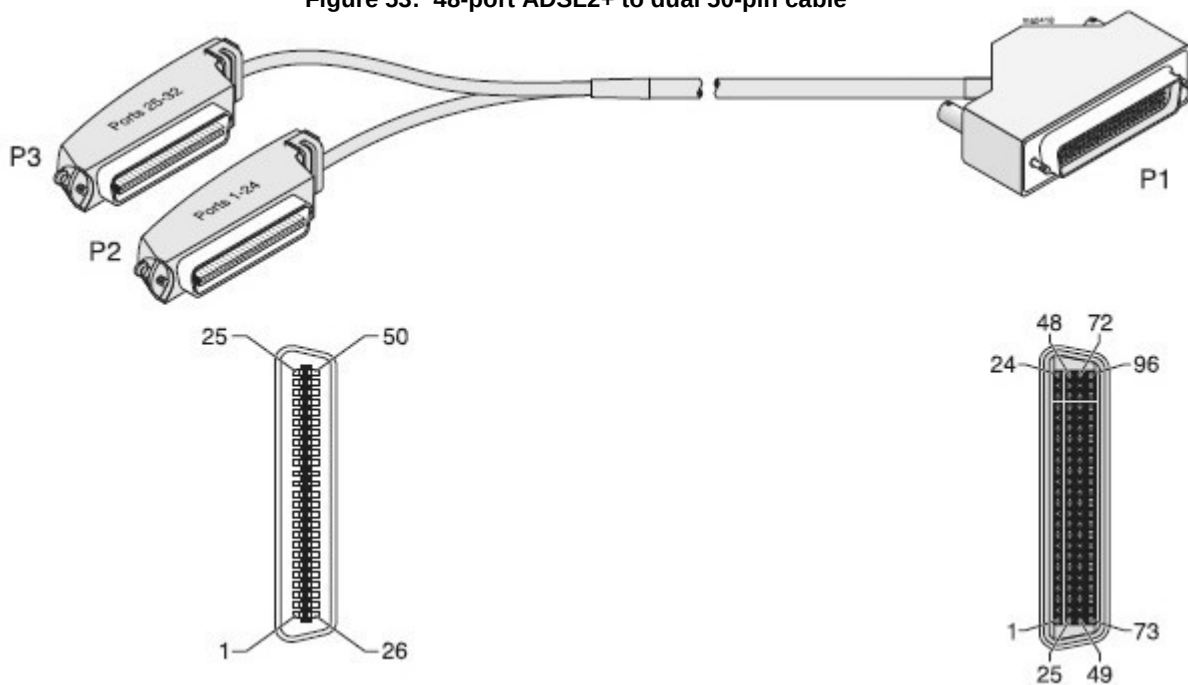


Table 58: 48-port ADSL to dual-50-pin cable pinouts

Pair	Signal	Color	From	To	Binder group
1	Tip	White/Blue	P1-2	P2-26	1 (Blue)
	Ring	Blue/White	P1-1	P2-1	
2	Tip	White/Orange	P1-4	P2-27	
	Ring	Orange/White	P1-3	P2-2	
3	Tip	White/Green	P1-6	P2-28	
	Ring	Green/White	P1-5	P2-3	
4	Tip	White/Brown	P1-8	P2-29	
	Ring	Brown/White	P1-7	P2-4	
5	Tip	White/Slate	P1-10	P2-30	
	Ring	Slate/White	P1-9	P2-5	
6	Tip	Red/Blue	P1-12	P2-31	
	Ring	Blue/Red	P1-11	P2-6	
7	Tip	Red/Orange	P1-14	P2-32	
	Ring	Orange/Red	P1-13	P2-7	
8	Tip	Red/Green	P1-16	P2-33	
	Ring	Green/Red	P1-15	P2-8	
9	Tip	Red/Brown	P1-18	P2-34	
	Ring	Brown/Red	P1-17	P2-9	
10	Tip	Red/Slate	P1-20	P2-35	
	Ring	Slate/Red	P1-19	P2-10	
11	Tip	Black/Blue	P1-22	P2-36	
	Ring	Blue/Black	P1-21	P2-11	
12	Tip	Black/Orange	P1-24	P2-37	
	Ring	Orange/Black	P1-23	P2-12	

Table 58: 48-port ADSL to dual-50-pin cable pinouts (Continued)

Pair	Signal	Color	From	To	Binder group
13	Tip	White/Blue	P1-26	P2-38	2 (Orange)
	Ring	Blue/White	P1-25	P2-13	
14	Tip	White/Orange	P1-28	P2-39	
	Ring	Orange/White	P1-27	P2-14	
15	Tip	White/Green	P1-30	P2-40	
	Ring	Green/White	P1-29	P2-15	
16	Tip	White/Brown	P1-32	P2-41	
	Ring	Brown/White	P1-31	P2-16	
17	Tip	White/Slate	P1-34	P2-42	
	Ring	Slate/White	P1-33	P2-17	
18	Tip	Red/Blue	P1-36	P2-43	
	Ring	Blue/Red	P1-35	P2-18	
19	Tip	Red/Orange	P1-38	P2-44	
	Ring	Orange/Red	P1-37	P2-19	
20	Tip	Red/Green	P1-40	P2-45	
	Ring	Green/Red	P1-39	P2-20	
21	Tip	Red/Brown	P1-42	P2-46	
	Ring	Brown/Red	P1-41	P2-21	
22	Tip	Red/Slate	P1-44	P2-47	
	Ring	Slate/Red	P1-43	P2-22	
23	Tip	Black/Blue	P1-46	P2-48	
	Ring	Blue/Black	P1-45	P2-23	
24	Tip	Black/Orange	P1-48	P2-49	
	Ring	Orange/Black	P1-47	P2-24	

Table 58: 48-port ADSL to dual-50-pin cable pinouts (Continued)

Pair	Signal	Color	From	To	Binder group
25	Tip	White/Blue	P1-50	P3-26	3 (Green)
	Ring	Blue/White	P1-49	P3-1	
26	Tip	White/Orange	P1-52	P3-27	
	Ring	Orange/White	P1-51	P3-2	
27	Tip	White/Green	P1-54	P3-28	
	Ring	Green/White	P1-53	P3-3	
28	Tip	White/Brown	P1-56	P3-29	
	Ring	Brown/White	P1-55	P3-4	
29	Tip	White/Slate	P1-58	P3-30	
	Ring	Slate/White	P1-57	P3-5	
30	Tip	Red/Blue	P1-60	P3-31	
	Ring	Blue/Red	P1-59	P3-6	
31	Tip	Red/Orange	P1-62	P3-32	
	Ring	Orange/Red	P1-61	P3-7	
32	Tip	Red/Green	P1-64	P3-33	
	Ring	Green/Red	P1-63	P3-8	
33	Tip	Red/Brown	P1-66	P3-34	
	Ring	Brown/Red	P1-65	P3-9	
34	Tip	Red/Slate	P1-68	P3-35	
	Ring	Slate/Red	P1-67	P3-10	
35	Tip	Black/Blue	P1-70	P3-36	
	Ring	Blue/Black	P1-69	P3-11	
36	Tip	Black/Orange	P1-72	P3-37	
	Ring	Orange/Black	P1-71	P3-12	

Table 58: 48-port ADSL to dual-50-pin cable pinouts (Continued)

Pair	Signal	Color	From	To	Binder group
37	Tip	White/Blue	P1-74	P3-38	4 (Brown)
	Ring	Blue/White	P1-73	P3-13	
38	Tip	White/Orange	P1-76	P3-39	
	Ring	Orange/White	P1-75	P3-14	
39	Tip	White/Green	P1-78	P3-40	
	Ring	Green/White	P1-77	P3-15	
40	Tip	White/Brown	P1-80	P3-41	
	Ring	Brown/White	P1-79	P3-16	
41	Tip	White/Slate	P1-82	P3-42	
	Ring	Slate/White	P1-81	P3-17	
42	Tip	Red/Blue	P1-84	P3-43	
	Ring	Blue/Red	P1-83	P3-18	
43	Tip	Red/Orange	P1-86	P3-44	
	Ring	Orange/Red	P1-85	P3-19	
44	Tip	Red/Green	P1-88	P3-45	
	Ring	Green/Red	P1-87	P3-20	
45	Tip	Red/Brown	P1-90	P3-46	
	Ring	Brown/Red	P1-89	P3-21	
46	Tip	Red/Slate	P1-92	P3-47	
	Ring	Slate/Red	P1-91	P3-22	
47	Tip	Black/Blue	P1-94	P3-48	
	Ring	Blue/Black	P1-93	P3-23	
48	Tip	Black/Orange	P1-96	P3-49	
	Ring	Orange/Black	P1-95	P3-24	

48-port ADSL to dual 50-pin connector cables

[Table 59](#) describes the 48-port ADSL2+ to dual 50-pin connector cables.

Table 59: 48-port ADSL to dual 50-pin connector cables

MXK CABLE PART NAME	DESCRIPTION
MALC-CBL-ADSL-48	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, 10 FT/ 3.05M
MALC-CBL-ADSL-48-10FTF	96PIN TO 2 50PIN FEMALE CONNECTOR, 10FT/3.05M
MALC-CBL-ADSL-48-15FT	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, 15 FT/ 4.57M
MALC-CBL-ADSL-48-30M	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, 98.4 FT/ 30M
MALC-CBL-ADSL-48-30MF	96 PIN TO (2) 50-PIN FEMALE CONNECTOR, 98.4FT/30M LENGTH
MALC-CBL-ADSL-48-4FT	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, 4 FT/1.22M
MALC-CBL-ADSL-48-50M	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, 164 FT/50M
MALC-CBL-ADSL-48-50FTF	96PIN TO 92) 50PIN CONNECTOR, 48-PORT CARDS, 50FT/ 15.24M
MALC-CBL-ADSL-48-70M	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, 229 FT/70M
MALC-CBL-ADSL-48-125FTF	96PIN TO (2) 50PIN CONNECTOR, 48-PORT CARDS, 125FT/38.1M
MALC-CBL-ADSL-48-90DEG-10FT	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, 90 DEGREES, 10 FT/3.05M
MALC-CBL-ADSL-48-90DEG-4FT	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, 90 DEGREES, 4 FT/1.22M
MALC-CBL-ADSL-48-UP-30FT	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, ROUTES UPWARDS, 30 FT/9.1M
MALC-CBL-ADSL-48-UP-60FT	96 PIN TO (2) 50-PIN CONNECTOR, 48-PORT CARDS, ROUTES UPWARDS, 60 FT/18.28M

Variations of 48-port ADSL to dual 50-pin connector cables

[Table60](#) lists variations of the 48-port ADSL2+ to dual 50-pin connector cables. These cables use the pinouts listed in [Table58 on page 601](#).

Table 60: Variations of 48-port ADSL to dual 50-pin connector cables

MXK CABLE PART NAME	DESCRIPTION
MALC-CBL-ADSL-48-100FT-BLUNT	96 PIN TO BLUNT END, 100 FT/30.5M
MALC-CBL-ADSL-48-350FT-BLUNT	96 PIN TO BLUNT END, 350 FT/106.7M

ADSL2+ testing (SELT/DELT) on the MXK

This section discusses:

- [SELT \(Single-End Loop Test\), page606](#)
- [DELT \(Dual-End Loop Test\), page611](#)

Single-End Loop Test (SELT) and Dual-End Loop Test (DELT) are loop tests which can be used to proactively pre-qualify a loop (SELT) or reactively test a loop after a modem has been deployed (DELT).



Note: Only one test (SELT, DELT, or TAC) can run at a time on an interface. Otherwise the current test will be interrupted by the newly configured test, thus the test results returned will be inaccurate.

Before starting a SELT/ DELT test on an interface, you can use the **sel| del| show status <interface>** command to identify whether there is a SELT or DELT test is running on this interface or not.



Note: Only one SELT test can run at a time on a board. But multiple DELT tests can run at same time on a board, as long as they are not running in the same interface.



Note: Test limitations:

- Test range is 600 to 9000 feet.
- Mixed gauge wire is not supported.
- Results have +/- 10% variance.

SELT (Single-End Loop Test)

SELT is a single-ended test. A copper loop can be tested from the MXK only, without the need for any external test equipment in either the CO or at the remote end of the loop. SELT is primarily used for proactive loop pre-qualification.

Starting SELT

The SELT algorithm estimate length and gauge of the wire connected to the interface. It also measures loop noise floor at each ADSL2+ subcarrier frequency. There must be no CPE connected to the line, even if it is turned off, although a phone maybe connected. Only one SELT test maybe run at a time.

Use the **sel| start <interface>** command to start a SELT test on an ADSL2+ interface. Make sure the administrative status on this ADSL2+ interface is down.

- 1 Set the administrative status of the ADSL2+ interface to down.

```
zSH> port down 1-4-3-0/ads1
```

1-4-3-0/adsl set to admin state DOWN

2 Start the SELT test on it.

```
zSH> selt start 1-4-3-0/adsl
Selt test started on interface 1-4-3-0/adsl
```

Showing SELT status

Use the **selt show status <interface>** command to display SELT test progress and loop characterization on an ADSL2+ interface.

[Table61](#) describes the SELT test status parameters.

Table 61: SELT test status parameters

Parameter	Description
status	The status of the current SELT test. Values: in-progress This SELT is currently running complete This SELT has been run at least once since system reset and has completed aborted This SELT was aborted stopped-in-progress The request to abort the test is still in progress not-started This SELT has not been run on this interface since last system reset results-cleared This SELT test results table was removed as requested. time-out This SELT was aborted automatically because it did not complete within the max-duration. unsupported The device or interface does not support SELT port-active The interface's administrative status has not been set to down as required. measurement invalid This SELT returns invalid measurements
max-duration	Maximum number of seconds the test is allowed to run. Default value is Disabled.
cfg-gauge	Configured expected wire gauge. Default value is awg 26.
cfg-cable	Configured expected cable type, real or simulated. Default value is real.
time-left	Estimated number of seconds remaining in the current test.
device	ADSL chip set in MXK used to perform SELT test.
bridge-taps	Whether or not this chip set can run SELT test in the presence of bridge taps.
date-time	Date and time that the most-recently run test completed.

Table 61: SELT test status parameters (Continued)

Parameter	Description
length	Estimated length of the loop.
gauge	Estimated wire gauge (if supported by hardware)

The following examples show the SELT status during the test and after the test completed:

```
zSH> selt show status 1-4-3-0/adsl
status:      in-progress
max-duration: disabled
cfg-gauge:   awg26
cfg-cable:   real
time-left:   307      seconds
device:      broadcom-6411
bridge-taps: not-supported
date-time:   no results have been generated
length:      unknown
gauge:       unknown
```

```
zSH> selt show status 1-4-3-0/adsl
status:      complete
max-duration: disabled
cfg-gauge:   awg26
cfg-cable:   real
time-left:   0 seconds
device:      broadcom-6411
bridge-taps: not-supported
date-time:   results generated 30 jun 2009, 22:35:32
length:      131 feet
gauge:       awg26
```

Showing SELT noise

Use the **selt show noise <interface> [start-index [num-vals]]** command to display the loop noise floor results of a SELT test. There is one noise measurement per ADSL2+ tone. Each noise value has units of dBm/Hz.

– <interface> can be in the form of ifIndex (e.g. 432), name/type (e.g. 1-4-1-0/adsl) or shelf/slot/port/subport/type (e.g. 1/4/1/0/adsl).

– [start-index]: (0..511) is the tone index with which to start. 0 is 4.3125 kHz, 1 is 8.625 kHz, up to 511 which is 2208.0000 kHz.

– [num-vals]: the number of tones to display.

```
zSH> selt show noise 1-4-3-0/adsl
Results generated 10 sep 2006, 14:35:56.
Tone   Tone Freq   Noise
Index  (kHz)         (dBm/Hz)
-----
```

0	4.3125	-95.7
1	8.6250	-118.3
2	12.9375	-121.4
3	17.2500	-123.8
4	21.5625	-124.9
5	25.8750	-126.3
6	30.1875	-125.5
7	34.5000	-121.8
8	38.8125	-113.6
9	43.1250	-125.9
10	47.4375	-127.7
11	51.7500	-128.4
12	56.0625	-128.3
13	60.3750	-128.5
14	64.6875	-128.3
15	69.0000	-124.4

[etc, up to index 511]

```
zSH> selt show noise 1-4-1-0/adsl 253 6
Results generated 10 sep 2006, 14:35:56.
Tone   Tone Freq   Noise
Index   (kHz)      (dBm/Hz)
-----
253    1095.3750    -122.0
254    1099.6875    -122.6
255    1104.0000    -121.9
256    1108.3125    no measurement
257    1112.6250    no measurement
258    1116.9375    no measurement
259    1121.2500    no measurement
```

Setting SELT

The MXK supports the following SELT commands to change the SELT settings.

- **selt set units <awg | metric | japan>**

Set the SELT display units for the chassis.

- awg: Show wire diameters and lengths in English units. This is default value.
- metric: Show wire diameters and lengths in Metric units.
- japan: Show wire diameters and lengths in Japanese metric units.

- **selt set max-duration <interface> <num-seconds>**

Set the maximum amount of time, in seconds, that a SELT test can run on the ADSL2+ interface. If a SELT test runs for more than <num-seconds> it will be aborted.

Setting max-duration to zero disables SELT timeout on an interface. By default, max-duration is Disabled.

Note that, in order to get valid results, a SELT test must be allowed to run to completion, and this may take several minutes, depending on the speed of the processor used to do the computation.

- **selt set gauge <interface> <wire-gauge>**

Set the expected diameter of the wire connected to an ADSL2+ interface. The diameter may be set using any units, regardless of the display units set with the **selt set units** command. The <wire-gauge> option must use one of these settings:

- unknown: unknown wire gauge
- awg19: 19 gauge
- awg22: 22 gauge
- awg24: 24 gauge
- awg26: 26 gauge. This is default value.
- 32mm: 0.32 millimeters
- 40mm - 0.40 millimeters
- 50mm - 0.50 millimeters
- 63mm - 0.63 millimeters
- 65mm - 0.65 millimeters
- 90mm - 0.90 millimeters

- **selt set cable <interface> <cable-type>**

Set the type of cable being tested, real or simulated.

<cable-type>: real, ds190, ds400. The *real* setting indicates that an actual physical cable is connected to the interface, and this is the default setting. In a lab or test environment, the cable may be simulated and use the *ds190* or *ds400* setting.

Examples:

```
zSH> selt set units metric
Selt information will be displayed in metric units.
```

```
zSH> selt set max-duration 1-4-3-0/adsl 200
Selt test timeout on interface 1-4-3-0/adsl set to 200 seconds.
```

Aborting SELT

Use the **selt abort <interface>** command to terminate a SELT test on an ADSL2+ interface. Note that it may take some time (perhaps as much as a minute) for the SELT test to abort.

```
zSH> selt abort 1-4-3-0/adsl
Selt test aborted on interface 1-4-3-0/adsl
```

Clearing stored DELT results

Use the **selt clear <interface>** command to clear stored results of a SELT on an ADSL2+ interface.

```
zSH> selt clear 1-4-3-0/adsl  
Selt results cleared on interface 1-4-3-0/adsl
```

DELT (Dual-End Loop Test)

DELT is a dual-ended test that requires equipment that supports the DELT feature at both ends of the copper loop. While this prevents DELT from being used on loops where no CPE has yet been deployed, DELT offers a deeper set of loop tests, and can provide very valuable information on the condition of a copper loop. DELT is primarily used for reactive tests on a loop after a modem has been deployed to either help troubleshoot a line or capture a baseline of loop characteristics at the time of installation in order to assist in future troubleshooting. In addition, DELT can assist in predetermining if there is sufficient capability on that line to support new services, such as voice and video.

Starting DELT

DELT requires that there be an endpoint on the line to be tested. The endpoint equipment must also support DELT (e.g. Zhone CPE 6212). DELT is expected to be used in situations where the line quality may not be good enough for the port to train normally. Therefore, when performing DELT, the devices on each end of the line communicate more slowly than usual. It may take several minutes for the devices to exchange DELT information. Once the devices have shared DELT information, the line will return to an idle state.

Use the **delt start <interface>** command to start a DELT test on an ADSL2+ interface. To run DELT commands, the interface does not have to be down.

```
zSH> delt start 1-4-1-0/adsl  
Delt test started on interface 1-4-1-0/adsl
```

Showing DELT status

Use the **delt show status <interface>** command to display DELT test progress on an ADSL2+ interface.

Table62 describes the DELT test status parameters.

Table 62: DELT test status parameters

Parameter	Description
Status	The status of the current DELT test. Values: in-progress The Loop Diagnostics process is in progress success The recent DELT command completed successfully aborted DELT was stopped, possible by the user none The DELT SNMP operation fails cannot-run Cannot start the test due to unspecified reason, possible the port is out of range illegal-mode The interface or line is in an illegal mode admin-up The interface has not been set administratively-down table-full DELT results table is full no-resource The system lacks a resource such as free memory
Device	The ADSL chip set in device used to perform the test.
Delt results generated <date>, <time>	The date and time of the test most-recently completed.
If the test is successful, the upstream and downstream values of the following parameters will be displayed:	
Attainable Bit Rate (bps)	Maximum Attainable Data Rate in bits per second. The maximum net data rate currently attainable by the ATU-C transmitter and ATU-R receiver (when referring to downstream direction) or by the ATU-R transmitter and ATU-C receiver (when referring to upstream direction).
Loop Attenuation (dB)	When referring to the downstream direction, it is the measured difference in the total power transmitted by the ATU-C and the total power received by the ATU-R over all sub-carriers during diagnostics mode. When referring to the upstream direction, it is the measured difference in the total power transmitted by the ATU-R and the total power received by the ATU-C over all sub-carriers during diagnostics mode. It ranges from 0 to 127 dB.
Signal Attenuation (dB)	When referring to the downstream direction, it is the measured difference in the total power transmitted by the ATU-C and the total power received by the ATU-R over all sub-carriers during SHOWTIME after the diagnostics mode. When referring to the upstream direction, it is the measured difference in the total power transmitted by the ATU-R and the total power received by the ATU-C over all sub carriers during SHOWTIME after the diagnostics mode. Range is 0 to 127 dB.

Table 62: DELT test status parameters (Continued)

Parameter	Description
SNR Margin (dB)	It is the maximum increase in dB of the noise power received at the ATU (ATU-R on downstream direction and ATU-C on upstream direction), such that bit error rate (BER) requirements are met for all bearer channels received at the ATU. It ranges from -64 to 63 dB.
Actual Transmit Power (dBm)	Actual Aggregate Transmit Power from the ATU (ATU-R on downstream direction and ATU-C on upstream direction), at the instant of measurement. It ranges from -31 to 31 dBm.

The following examples show the DELT status during the test and after the test completed:

```
zSH> delt show status 1-4-1-0/adsl
Status: in-progress
Device: broadcom-6411
No delt results have been generated on this interface.
```

```
zSH> delt show status 1-4-1-0/adsl
Status: success
Device: broadcom-6411
Delt results generated 30 jun 2009, 22:47:51.
      Downstream      Upstream
      -----      -
Attainable Bit Rate (bps)      23032000      10800000
Loop Attenuation (dB)          0.0          1.2
Signal Attenuation (dB)        0.0          0.2
SNR Margin (dB)                6.1          6.0
Actual Transmit Power (dBm)    16.1          11.3
```

Showing DELT noise

Use the **delt show noise <interface> [start-index [num-vals]]** command to display DELT noise, attenuation, and SNR per subcarrier.

The following describes the command options:

- <interface> can be in the form of ifIndex (e.g. 432), name/type (e.g. 1-4-1-0/adsl) or shelf/slot/port/subport/type (e.g. 1/4/1/0/adsl).
- [start-index: (0..511)] is the tone index with which to start.
- [num-vals]: the number of tones to display

[Table 63](#) describes the parameters in the DELT show noise command result.

Table 63: DELT show noise parameters

Parameter	Description
Tone Index	Tone index. In the range of 0 to 511. 0 indicates 4.3125 kHz Tone Freq, 1 indicates 8.625 kHz, ..., 511 indicates 2208.0000 kHz.
Tone Freq (kHz)	Tone frequency. Tone Freq = (Tone Index + 1) x 4.3125 kHz.

Table 63: DELT show noise parameters (Continued)

Parameter	Description
If the test is successful, the upstream and downstream values of the following parameters will be displayed:	
Attenuation (dB)	An listing of up to 512 real H(f) logarithmic representation values in dB for the respective transmission direction. It supports up to 512 (downstream) sub-carriers. The number of utilized values for a direction (downstream or upstream) depends on the number of sub-carriers defined for that direction (NSCds or NSCus). Each row in the table contains a pair of Hlog(f = i*Df) values for a particular sub-carrier index. The real Hlog(f) value has a range of -96.3 to 6.0 dB. "no data" indicates that no measurement could be done for the sub-carrier because it is out of the passband or that the attenuation is out of range to be represented.
Noise (dBm/Hz)	An listing of up to 512 real Quiet Line Noise values in dBm/Hz for the respective transmission direction. Like Attenuation, it supports up to 512 (downstream) sub-carriers. Each row in the table contains a pair of QLN(f= i*Df) value values for a particular sub-carrier index. The QLN(f) value has a range of -150.0 to -23.0 dBm/Hz. "no data" indicates that no measurement could be done for the sub-carrier because it is out of the passband or that the noise PSD is out of range to be represented.
Signal-to-noise ratio (dB)	This is the SNR Margin per sub-carrier, expressing the ratio between the received signal power and received noise power per subscriber. It is a listing of 512 values. Like Attenuation, it supports up to 512 (downstream) sub-carriers. The SNR value has a range of -32 to 95 dB. "no data" indicates that no measurement could be done for the sub-carrier, probably because it is out of the PSD mask passband or that the noise PSD is out of range to be represented.

```
zSH> delt show noise 1-4-2-0/adsl
```

```
Delt results generated 30 jun 2009, 22:47:51.
```

Tone Index	Tone Freq (kHz)	Attenuation (dB)		Noise (dBm/Hz)		SNR (dB)	
		dnstream	upstream	dnstream	upstream	dnstream	upstream
0	4.3125	-78.1	-25.8	-84.0	-146.0	0.0	no data
1	8.6250	-40.7	-68.6	-127.0	-130.0	0.0	no data
2	12.9375	-42.9	-71.7	-127.0	-130.5	0.0	no data
3	17.2500	-42.9	-71.7	-127.0	-130.5	0.0	no data
4	21.5625	-45.3	-71.7	-127.0	-129.5	0.0	no data
5	25.8750	-48.7	-66.9	-127.0	-128.5	0.0	no data
6	30.1875	-48.7	-20.7	-127.0	-126.0	0.0	no data
7	34.5000	-42.6	-10.0	-127.0	-120.0	0.0	29.5
8	38.8125	-45.3	-0.6	-127.0	-113.5	0.0	35.5
9	43.1250	-51.2	1.5	-127.0	-113.0	0.0	41.0
10	47.4375	-51.7	1.7	-127.0	-112.0	0.0	43.5
11	51.7500	-54.3	2.3	-127.0	-113.5	0.0	46.0
12	56.0625	-51.2	2.9	-127.0	-114.5	0.0	48.0
13	60.3750	-54.3	3.2	-127.0	-111.5	0.0	49.0
14	64.6875	-47.6	3.2	-127.0	-112.5	0.0	51.0

```

15      69.0000      -47.7      2.9      -127.0      -110.5      0.0      51.5
[etc, up to index 511]

```

```
zSH> delt show noise 1-4-1-0/adsl 7 6
```

```
Delt results generated 30 jun 2009, 22:47:51.
```

Tone Index	Tone Freq (kHz)	Attenuation (dB)		Noise (dBm/Hz)		SNR (dB)	
		dnstream	upstream	dnstream	upstream	dnstream	upstream
7	34.5000	-42.6	-10.0	-127.0	-120.0	0.0	29.5
8	38.8125	-45.3	-0.6	-127.0	-113.5	0.0	35.5
9	43.1250	-51.2	1.5	-127.0	-113.0	0.0	41.0
10	47.4375	-51.7	1.7	-127.0	-112.0	0.0	43.5
11	51.7500	-54.3	2.3	-127.0	-113.5	0.0	46.0
12	56.0625	-51.2	2.9	-127.0	-114.5	0.0	48.0
13	60.3750	-54.3	3.2	-127.0	-111.5	0.0	49.0

Aborting DELT

Use the **delt abort <interface>** command to terminate a DELT test on an ADSL2+ interface.

Abort this test.

```
zSH> delt abort 1-4-1-0/adsl
```

```
Delt test aborted on interface 1-4-1-0/adsl
```

Clearing stored DELT results

Use the **delt clear <interface>** command to clear the stored DELT results on an ADSL2+ interface.

Clear the last DELT test results on the ADSL2+ interface.

```
zSH> delt clear 1-4-1-0/adsl
```

```
Delt results cleared on interface 1-4-1-0/adsl
```


13 MXK 24-PORT VDSL2 CARD

This chapter describes the MXK-VDSL2-BCM-17A-24 card and VDSL2 configuration:

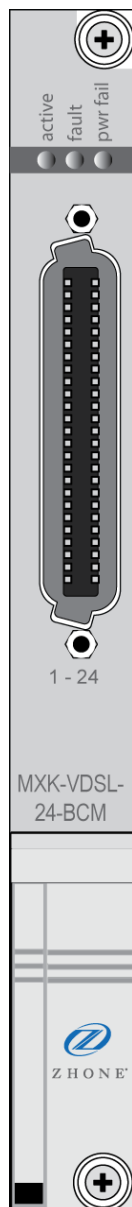
- [VDSL2 24-port single slot card, page618](#)
- [VDSL2 interfaces, page622](#)
- [VDSL2 statistics, page636](#)
- [VDSL2 24 port card pinouts, page645](#)

VDSL2 24-port single slot card

This section describes the MXK-VDSL2-BCM-17A-24 single-slot card and VDSL2 card configuration:

- [VDSL2 card overview, page618](#)
- [VDSL2 card specifications, page619](#)
- [VDSL2 24-port card configuration, page619](#)
- [View additional card information, page622](#)

VDSL2 card overview



The MXK-VDSL2-24-BCM is a single-slot 24-port VDSL2 subscriber line card, which provides high symmetric and asymmetric bandwidth and supports up to 17a profile.

The MXK-VDSL2-24-BCM card can be used with the Zhone VDSL2 CPE devices. This architecture allows VDSL2 users to access the maximum bandwidth available over twisted-pair, copper phone lines.

VDSL2 card specifications

Table 64: MXK-VDSL2-24 card specifications

Specification	Value
Size	Single slot
Density	24 ports
Physical interfaces	One (1) RJ-21X Champ 50-pin connector
Nominal line rates	Fast mode: upstream: 58.177 Mbps downstream: 100.160 Mbps Interleaved mode (minINP[2], maxDelay [20]): upstream: 29.835 Mbps downstream: 100.205 Mbps PhyR™ mode: upstream: 18.232 Mbps downstream: 73.44 Mbps
Power consumption	52.5W nominal All ports trained 60.0W maximum All ports trained 30W nominal No ports trained 36W maximum No ports trained 1W per active port
Standards supported	ITU G.993.2 VDSL 2 ITU G.993.1 VDSL

VDSL2 24-port card configuration

Each card in the system must have a **card-profile**. The line card type determines the parameter settings in the **card-profile** and the software image for the card. Performing a **card add <slot #>** automatically creates the **card-profile** for the card with the correct software image and settings.

Table 65 provides the type and the software image for the VDSL2 card on the MXK.

Table 65: MALC-VDSL2-24 card type and software image

Card	Type	Name of software image
MXK-VDSL2-24	10210	<i>mxlc24vdsl2.bin</i>

Configuring a VDSL2 card

To configure a VDSL2 card on the MXK:

- 1 Install a VDSL2 card in the desired line card slot.
- 2 Create a card-profile for the card.

```
zSH> card add 1
```

After performing a **card add** in a slot, the slot resets and begins downloading the software image from the flash card. This could take a few moments.

When the card has finished loading, a log message similar to the following is displayed (if logging is enabled):

```
zSH> Card in slot slot-number changed state to RUNNING
```

- 3 Verify the card by entering **slots**:

```
zSH> slots
Uplinks
a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
1: MXK 24 PORT VDSL2 (RUNNING)
5: MXK ADSL-48-B Bonded (RUNNING)
6: MXK ADSL-48-B Bonded (RUNNING)
```

- 4 Verify the **card-profile** for the VDSL2 card:

```
zSH> get card-profile 1/1/10210
card-profile 1/1/10210
sw-file-name: -----> {mxlc24vdsl2.bin}
admin-status: -----> {operational}
upgrade-sw-file-name: ---> {}
upgrade-vers: -----> {}
admin-status-enable: ----> {enable}
sw-upgrade-admin: -----> {reloadcurrrev}
sw-enable: -----> {true}
sw-upgrade-enable: -----> {false}
card-group-id: -----> {0}
hold-active: -----> {false}
weight: -----> {nopreference}
card-line-type: -----> {unknown-type}
card-atm-configuration: -> {notapplicable}
```

```

card-line-voltage: -----> {not-used}
maxvpi-maxvci: -----> {notapplicable}
card-init-string: -----> {}
wetting-current: -----> {disabled}
pwe-timing-mode: -----> {none}

```

- 5 View card information including the state of the card and how long the card has been running:

```

zSH> slots 1
Type           : MXK 24 PORT VDSL2
Card Version   : 00001
EEPROM Version : 1
Serial #       : 2460301
CLEI Code      : No CLEI
Card-Profile ID : 1/1/10210
Shelf          : 1
Slot           : 1
ROM Version    : MXK 2.0.100
Software Version: MXK 2.0.117
State          : RUNNING
Mode           : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat  : 102
Fault reset    : enabled
Uptime         : 6 hours, 7 minutes

```

Verifying the slot card installation

After saving the **card-profile** record, the slot card in that slot resets and downloads the software image from the flash card. This could take a few moments.

When the card finishes loading, a log message similar to the following is displayed (if logging is enabled):

```
zSH> Card in slot slot-number changed state to RUNNING
```

You can also use the **slots** command with the slot number of the card to view the state of the card and other information.

```

zSH> slots 1
Type           : MXK 24 PORT VDSL2
Card Version   : 00001
EEPROM Version : 1
Serial #       : 2460301
CLEI Code      : No CLEI
Card-Profile ID : 1/1/10210
Shelf          : 1
Slot           : 1
ROM Version    : MXK 2.0.100
Software Version: MXK 2.0.117
State          : RUNNING
Mode           : FUNCTIONAL
Heartbeat check : enabled

```

```

Longest hbeat    : 102
Fault reset     : enabled
Uptime          : 6 hours, 7 minutes

```

To view the status of all the cards, use the **slots** command without any arguments:

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  1: MXK 24 PORT VDSL2 (RUNNING)
  5: MXK ADSL-48-B Bonded (RUNNING)
  6: MXK ADSL-48-B Bonded (RUNNING)

```

View additional card information

View the EPROM version of the card:

```

zSH> eeshow card 1
EEPROM contents: for slot 1
EEPROM_ID    : 00 -- CARD
Version      : 01
Size         : 054
CardType     : 10210 -- MXLC24VDSL2
CardVersion  : 00001
SerialNum    : 02460301
ShelfNumber  : 00001
CLEI Code    : No CLEI
Cksum       : 0x2922

```

View the EPROM version of the daughter card:

```

zSH> eeshow 1d 1
EEPROM contents: for slot 1
EEPROM_ID    : 01 -- 1DAUGHTER
Version      : 01
Size         : 022
CardType     : 10210 -- MXLC24VDSL2
CardVersion  : 00001
SerialNum    : 02460317
ShelfNumber  : 00001
CLEI Code    : No CLEI
Cksum       : 0x2812

```

VDSL2 interfaces

This section covers:

- [VDSL interface profiles, page623](#)
- [vdsl-config default parameters, page623](#)

- [vdsl-co-config default parameters, page627](#)
- [View vdsl-cpe-config profile default parameters, page630](#)
- [Configure VDSL2 profiles to cap train rates, page634](#)
- [Configure Broadcom Phy-R™, page634](#)

VDSL interface profiles

VDSL2 interface configuration consists of three profiles:

- **vdsl-config**
- **vdsl-co-config**
- **vdsl-cpe-config**

vdsl-config default parameters

The parameter defaults set in the **vdsl-config** profile are appropriate for most configurations. When necessary, the **vdsl-config** parameters can be updated.

Viewing vdsl-config profile defaults

- 1 Verify the VDSL2 cards.
- 2 View the **vdsl-config** parameters and values.

```
zSH> show vdsl-config
transmit-mode:-----> autonegotiatemode vdslmode vdsl2mode
adsl2plusmode adsl2mode gdmtdmode
line-type:-----> fastonly interleavedonly
vdsl2-profile:-----> g993-2-8a g993-2-8b g993-2-8c
g993-2-8d g993-2-12a g993-2-12b g993-2-17a g993-2-30a
adslAnnexMModeEnabled:-----> true false
adslAnnexMPsdMask:-----> eu64 eu60 eu56 eu52 eu48 eu44 eu40
eu36 eu32
trellis-enabled:-----> true false
rs-enabled:-----> true false
psd-shape:-----> region-a-nus0 region-a-eu-32
region-a-eu-36 region-a-eu-40 region-a-eu-44 region-a-eu-48
region-a-eu-52 region-a-eu-56 region-a-eu-60 region-a-eu-64
region-a-eu-128 region-a-adlu-32 region-a-adlu-36 region-a-adlu-40
region-a-adlu-44 region-a-adlu-48 region-a-adlu-52 region-a-adlu-56
region-a-adlu-60 region-a-adlu-54 region-a-adlu-128 region-b-998-m1x-a
region-b-998-m1x-b region-b-998-m1x-nus0 region-b-998-m2x-a
region-b-998-m2x-m region-b-998-m2x-b region-b-998-m2x-nus0
region-b-998-e17-m2x-nus0 region-b-998-e17-m2x-nus0-m
region-b-998-ade17-m2x-nus0-m region-b-998-ade17-m2x-a
region-b-998-ade17-m2x-b region-b-998-e30-m2x-nus0
region-b-998-e30-m2x-nus0-m region-b-998-ade30-m2x-nus0-m
```

```
region-b-998-ade30-m2x-nus0-a region-b-997-m1c-a-7 region-b-997-m1x-m-8
region-b-997-m1x-m region-b-997-m2x-m-8 region-b-997-m2x-a
region-b-997-m2x-m region-b-997-hpe17-m1-nus0 region-b-997-hpe30-m1-nus0
region-b-997-e17-m2x-a region-b-997-e30-m2x-nus0
```

3 View the current **vdsl-config** default parameters for a VDSL2 port.

```
zSH> get vdsl-config 1-6-1-0/vdsl
vdsl-config 1-6-1-0/vdsl
transmit-mode: -----> {autonegotiatemode}
line-type: -----> {interleavedonly}
vdsl2-profile: -----> {g993-2-17a}
adslAnnexMModeEnabled: -----> {false}
adslAnnexMPsdMask: -----> {eu32}
trellis-enabled: -----> {true}
rs-enabled: -----> {true}
psd-shape: -----> {region-a-eu-32}
```

[Table66](#) defines **vdsl-config** profile parameters.

Table 66: vdsl-config parameters

Parameter	Definition
transmit-mode	The VDSL2 transmission standard to be used for the line. Supported values are: autoNegotiateMode: automatically negotiates all supported transmission modes. vdslMode: The VDSL standards supported are: standard long-reach-8k r8k std-8k lr-8k std-lr std-lr-8k vdsl2Mode: The VDSL2 standards supported are: g993-2-8a(1), g993-2-8b g993-2-8c g993-2-8d g993-2-12a g993-2-12b g993-2-17a g993-2-30a adsl2Mode: The modem negotiates rates up to G.992.3 and G.992.4 (ADSL2). adsl2PlusMode: The modem negotiates rates up to G.992.5 (ADSL2+). gdmtdMode: G.dmt Default: autoNegotiateMode
line-type	Specifies the channelization of the VDSL2 line. fastonly No impulse noise protection, but lowest possible latency. Recommended only where lowest possible latency is required (for example, gaming). interleaveonly Better impulse noise protection with higher latency. Recommended for all voice, video, and/or data deployments. Default: interleaveonly

Table 66: vdsl-config parameters (Continued)

Parameter	Definition
vdsl2-profile	The VDSL2 standard to be used for the line. Standards: g993-2-8a g993-2-8b g993-2-8c g993-2-8d g993-2-12a g993-2-12b g993-2-17a g993-2-30a Default: g993-2-17a
adslAnnexMModeEnabled	G.992.3/4 with Annex M true false Default: false
adslAnnexMPsdMask	Specifies the maximum transmit PSD allowed in the downstream channel on an BIS Annex-M line. eu64 eu60 eu56 eu52 eu48 eu44 eu40 eu36 eu32 Default: eu32
trellis-enabled	Enables/disables Trellis coding. true false Default: true
rs-enabled	Enables or disables RS coding. true false Default: true

Table 66: vdsl-config parameters (Continued)

Parameter	Definition
psd-shape	Maximum transmit PSD allowed on a VDSL2 line. region-a-nus0, region-a-eu-32, region-a-eu-36, region-a-eu-40, region-a-eu-44, region-a-eu-48, region-a-eu-52, region-a-eu-56, region-a-eu-60, region-a-eu-64, region-a-eu-128, region-a-adlu-32, region-a-adlu-36, region-a-adlu-40, region-a-adlu-44, region-a-adlu-48, region-a-adlu-52, region-a-adlu-56, region-a-adlu-60, region-a-adlu-64, region-a-adlu-128, region-b-998-m1x-a, region-b-998-m1x-b, region-b-998-m1x-nus0, region-b-998-m2x-a, region-b-998-m2x-m, region-b-998-m2x-b, region-b-998-m2x-nus0, region-b-998-e17-m2x-nus0, region-b-998-e17-m2x-nus0-m, region-b-998-ade17-m2x-nus0-m, region-b-998-ade17-m2x-a, region-b-998-ade17-m2x-b, region-b-998-e30-m2x-nus0, region-b-998-e30-m2x-nus0-m, region-b-998-ade30-m2x-nus0-m, region-b-998-ade30-m2x-nus0-a, region-b-997-m1c-a-7, region-b-997-m1x-m-8, region-b-997-m1x-m, region-b-997-m2x-m-8, region-b-997-m2x-a, region-b-997-m2x-m, region-b-997-hpe17-m1-nus0, region-b-997-hpe30-m1-nus0, region-b-997-e17-m2x-a, region-b-997-e30-m2x-nus0 Default: region-a-eu-32

vdsl-co-config default parameters

The VDSL2 downstream interface is the **vdsl-co-config** profile which defines downstream behavior.

Viewing vdsl-co-config profile defaults

View the **vdsl-co-config** parameters and their default values.

- 1 Verify the VDSL2 cards.

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  1: MXK 24 PORT VDSL2 (RUNNING)
  5: MXK ADSL-48-B Bonded (RUNNING)
  6: MXK ADSL-48-B Bonded (RUNNING)

```

- 2 View the **vdsl-co-config** parameters and values.

```

zSH> show vdsl-co-config
fastMaxTxRate:-----> {0 - 200000}
fastMinTxRate:-----> {0 - 200000}
interleaveMaxTxRate:-----> {0 - 200000}
interleaveMinTxRate:-----> {0 - 200000}
rateMode:-----> manual adapt-at-init dynamic
maxPower:-----> {-50 - 200}

```

```

maxSnrMgn:-----> {0 - 310}
minSnrMgn:-----> {0 - 310}
targetSnrMgn:-----> {0 - 310}
downshiftSnrMgn:-----> {0 - 310}
upshiftSnrMgn:-----> {0 - 310}
minDownshiftTime:-----> {0 - 16383}
minUpshiftTime:-----> {0 - 16383}
bitSwap:-----> disabled enabled
minINP:-----> noprotection halfsymbol singlesymbol
twosymbols threesymbols foursymbols fivesymbols sixsymbols sevensymbols
eightsymbols ninesymbols tensymbols elevensymbols twelvesymbols
thirteensymbols fourteensymbols fifteensymbols sixteensymbols
maxInterleaveDelay:-----> {0 - 63}
phyRSupport:-----> enable disable
phyRmaxINP:-----> {0 - 160}
phyRminRSoverhead:-----> {0 - 255}
phyRRtxRatio:-----> {0 - 255}

```

3 View the current **vdsl-co-config** default parameters for a VDSL2 port.

```

zSH> get vdsl-co-config 1-1-1-0/vdsl
vdsl-co-config 1-1-1-0/vdsl
fastMaxTxRate: -----> {200000}
fastMinTxRate: -----> {0}
interleaveMaxTxRate: -----> {200000}
interleaveMinTxRate: -----> {0}
rateMode: -----> {dynamic}
maxPower: -----> {200}
maxSnrMgn: -----> {127}
minSnrMgn: -----> {0}
targetSnrMgn: -----> {60}
downshiftSnrMgn: -----> {30}
upshiftSnrMgn: -----> {90}
minDownshiftTime: -----> {30}
minUpshiftTime: -----> {30}
bitSwap: -----> {enabled}
minINP: -----> {twosymbols}
maxInterleaveDelay: -----> {20}
phyRSupport: -----> {disable}
phyRmaxINP: -----> {0}
phyRminRSoverhead: -----> {0}
phyRRtxRatio: -----> {0}

```

[Table67](#) defines the parameter values for the **vdsl-co-config** profile.

Table 67: vdsl-co-config parameter definitions

Parameter	Description
fastMaxTxRate	Specifies the maximum downstream fast channel data rate in steps of 1000 bits/second. Default: 200,000

Table 67: vdsl-co-config parameter definitions (Continued)

Parameter	Description
fastMinTxRate	Specifies the minimum downstream fast channel data rate in steps of 1000 bits/second. Default: 0
interleaveMaxTxRate	Specifies the maximum downstream slow channel data rate in steps of 1000 bits/second. The maximum aggregate downstream transmit speed of the line can be derived from the sum of maximum downstream fast and slow channel data rates. Default: 200,000
interleaveMinTxRate	Specifies the minimum downstream slow channel data rate in steps of 1000 bits/second. The minimum aggregate downstream transmit speed of the line can be derived from the sum of minimum downstream fast and slow channel data rates. Default: 0
rateMode	Specifies the rate selection behavior for the line in the downstream direction. manual: forces the rate to the configured rate adapt-at-init: adapts the line at initialization only dynamic: adapts the line at initialization and showtime Default: dynamic
maxPower	Specifies the maximum aggregate downstream power level in the range 0 to 14.5 dBm. Default: 200
maxSnrMgn	Specifies the maximum downstream Signal/Noise Margin in units of 0.10 dB, for a range of 0 to 31.0 dB. Default: 127
minSnrMgn	Specifies the minimum downstream Signal/Noise Margin in units of 0.10 dB, for a range of 0 to 31.0 dB. Default: 0
targetSnrMgn	Specifies the target downstream Signal/Noise Margin in units of 0.10 dB, for a range of 0 to 31.0 dB. This is the Noise Margin the transceivers must achieve with a BER of 10^{-7} or better to successfully complete initialization. Default: 60
downshiftSnrMgn	Configured Signal/Noise Margin for rate downshift. If the noise margin falls below this level, the modem should attempt to decrease its transmit rate. Default: 30
upshiftSnrMgn	Configured Signal/Noise Margin for rate upshift. If the noise margin rises above this level, the modem should attempt to increase its transmit rate. Default: 90

Table 67: vdsl-co-config parameter definitions (Continued)

Parameter	Description
minDownshiftTime	Minimum time in seconds that the current margin is below downshiftSnrMgn before a downshift occurs. Default: 30
minUpshiftTime	Minimum time in seconds that the current margin is above upshiftSnrMgn before an upshift occurs. Default: 30
bitSwap	Enable or disable downstream bit swap. Default: enabled
minINP	The minimum impulse noise protection for the downstream bearer channel expressed in symbols. One symbol equals 250 uS. noProtection, halfSymbol, singleSymbol, twoSymbols, threeSymbols, fourSymbols, fiveSymbols, sixSymbols, sevenSymbols, eightSymbols, nineSymbols, tenSymbols, elevenSymbols, twelveSymbols, thirteenSymbols, fourteenSymbols, fifteenSymbols, sixteenSymbols Default: twoSymbols
maxInterleaveDelay	Specifies the maximum interleave delay for the downstream slow channel. Default: 20
phyRSupport	Enable or disable downstream PHYR. enable disable Default: disable
phyRmaxINP	PHYR maximum downstream impulse noise protection. A value of 0 specifies no protection. The values 5 through 160 specify the number of symbols in 1/10 increments. Default: 0
phyRminRSoverhead	PHYR minimum downstream RS overhead. Default: 0
phyRRtxRatio	PHYR minimum downstream fraction of the line rate allocated for retransmission. Default: 0

View vdsl-cpe-config profile default parameters

The VDSL2 upstream interface is the **vdsl-cpe-config** profile which defines upstream behavior.

Viewing vdsl-cpe-config profile defaults

View the **vdsl-cpe-config** parameters and their default values.

1 Verify the VDSL2 cards.

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
  b: MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  1: MXK 24 PORT VDSL2 (RUNNING)
  5: MXK ADSL-48-B Bonded (RUNNING)
  6: MXK ADSL-48-B Bonded (RUNNING)

```

2 View the **vdsl-cpe-config** parameters and values.

```

zSH> show vdsl-cpe-config
fastMaxTxRate:-----> {0 - 200000}
fastMinTxRate:-----> {0 - 200000}
interleaveMaxTxRate:-----> {0 - 200000}
interleaveMinTxRate:-----> {0 - 200000}
rateMode:-----> manual adapt-at-init dynamic
maxPower:-----> {-130 - 200}
maxSnrMgn:-----> {0 - 310}
minSnrMgn:-----> {0 - 310}
targetSnrMgn:-----> {0 - 310}
downshiftSnrMgn:-----> {0 - 310}
upshiftSnrMgn:-----> {0 - 310}
minDownshiftTime:-----> {0 - 16383}
minUpshiftTime:-----> {0 - 16383}
bitSwap:-----> disabled enabled
minINP:-----> noprotection halfsymbol singlesymbol
twosymbols foursymbols eightsymbols sixteensymbols
maxInterleaveDelay:-----> {0 - 63}
phyRSupport:-----> enable disable
phyRmaxINP:-----> {0 - 160}
phyRminRSoverhead:-----> {0 - 255}
phyRRtxRatio:-----> {0 - 255}

```

3 View the current **vdsl-cpe-config** default parameters for a VDSL2 port.

```

zSH> get vdsl-cpe-config 1-1-1-0/vdsl
vdsl-cpe-config 1-1-1-0/vdsl
fastMaxTxRate: -----> {200000}
fastMinTxRate: -----> {0}
interleaveMaxTxRate: -----> {200000}
interleaveMinTxRate: -----> {0}
rateMode: -----> {dynamic}
maxPower: -----> {130}
maxSnrMgn: -----> {127}
minSnrMgn: -----> {0}
targetSnrMgn: -----> {60}
downshiftSnrMgn: -----> {30}
upshiftSnrMgn: -----> {90}
minDownshiftTime: -----> {30}
minUpshiftTime: -----> {30}
bitSwap: -----> {enabled}

```

```

minINP: -----> {twosymbols}
maxInterleaveDelay: -----> {20}
phyRSupport: -----> {disable}
phyRmaxINP: -----> {0}
phyRminRSoverhead: -----> {0}
phyRRtxRatio: -----> {0}

```

[Table68](#) defines the parameter values for the **vdsl-cpe-config** profile.

Table 68: vdsl-cpe-config parameter definitions

Parameter	Definition
fastMaxTxRate	Specifies the maximum upstream fast channel data rate in steps of 1000 bits/second. The maximum aggregate upstream transmit speed of the line can be derived from the sum of maximum upstream fast and slow channel data rates. Default: 200,000
fastMinTxRate	Specifies the minimum upstream fast channel data rate in steps of 1000 bits/second. The minimum aggregate upstream transmit speed of the line can be derived from the sum of minimum upstream fast and slow channel data rates. Default: 0
interleaveMaxTxRate	Specifies the maximum upstream slow channel data rate in steps of 1000 bits/second. Default: 200,000
interleaveMinTxRate	Specifies the minimum upstream slow channel data rate in steps of 1000 bits/second. Default: 0
rateMode	Specifies the rate selection behavior for the line in the upstream direction. manual : forces the rate to the configured rate adapt-at-init : adapts the line at initialization only dynamic : adapts the line at initialization and showtime Default: dynamic
maxPower	Specifies the maximum aggregate upstream power level in the range 0 to 14.5 dBm. Default: 130
maxSnrMgn	Specifies the maximum upstream Signal/Noise Margin in units of 0.10 dB, for a range of 0 to 31.0 dB. Default: 127
minSnrMgn	Specifies the minimum upstream Signal/Noise Margin in units of 0.10 dB, for a range of 0 to 31.0 dB. Default: 0

Table 68: vdsl-cpe-config parameter definitions (Continued)

Parameter	Definition
targetSnrMgn	Specifies the target upstream Signal/Noise Margin in units of 0.10 dB, for a range of 0 to 31.0 dB. This is the Noise Margin the transceivers must achieve with a BER of 10^{-7} or better to successfully complete initialization. Default: 60
downshiftSnrMgn	Configured Signal/Noise Margin for rate downshift. If the noise margin falls below this level, the modem should attempt to decrease its transmit rate. Default: 30
upshiftSnrMgn	Configured Signal/Noise Margin for rate upshift. If the noise margin rises above this level, the modem should attempt to increase its transmit rate. Default: 90
minDownshiftTime	Minimum time that the current margin is below DownshiftSnrMgn before a downshift occurs. Default: 30
minUpshiftTime	Minimum time that the current margin is above UpshiftSnrMgn before an upshift occurs. Default: 30
bitSwap	Enable or disable upstream bit swap. enabled disabled Default: enabled
minINP	The minimum impulse noise protection for the upstream bearer channel expressed in symbols. One symbol equals 250 uS. noProtection, halfSymbol, singleSymbol, twoSymbols, fourSymbols, eightSymbols, sixteenSymbols Default: twoSymbols
maxInterleaveDelay	The maximum interleaving delay in the upstream direction. A value of zero indicates no delay introduced. Default: 20
phyRSupport	Enable or disable upstream PHYR. Default: disable
phyRmaxINP	PHYR maximum upstream impulse noise protection. A value of 0 specifies no protection. The values 5 through 160 specify the number of symbols in 1/10 increments. Default: 0
phyRminRSoverhead	PHYR minimum upstream RS overhead. Default: 0

Table 68: vdsl-cpe-config parameter definitions (Continued)

Parameter	Definition
phyRRtxRatio	PHYR minimum upstream fraction of the line rate allocated for retransmission. Default: 0

Configure VDSL2 profiles to cap train rates

You can control upstream and downstream train rates in Kbps for *fast* or *interleaved* modes in the **vdsl-config**, **vdsl-co-config**, and **vdsl-cpe-config** profiles.

[Table 69](#) shows the profiles and default parameters for upstream and downstream train rates.

Table 69: Profiles and parameters for capping upstream and downstream train rates

Profile	Parameter and train rates
vdsl-config	line-type: <i>fastonly</i> or <i>interleaveonly</i>
vdsl-co-config	fastMaxTxRate: {200000} fastMinTxRate: {0} or interleaveMaxTxRate: {200000} interleaveMinTxRate: {0}
vdsl-cpe-config	fastMaxTxRate: {200000} fastMinTxRate: {0} or interleaveMaxTxRate: {200000} interleaveMinTxRate: {0}

Configure Broadcom Phy-R™

Setting the Broadcom Phy-R™ parameters in the CO and CPE VDSL2 profiles is for advanced users.



Note: The Phy-R™ parameter in the VDSL2 CO profile cannot be used unless there is a Broadcom CPE modem at the customer site with Phy-R™ parameters in the VDSL2 CPE profile.

Enabling Phy-R™ parameters

Update the **vdsl-co-config** and the **vdsl-cpe-config**:

```
zSH> update vdsl-co-config 1/1/1
vdsl-co-config 1/1/1
```

```

Please provide the following: [q]uit.
fastMaxTxRate: -----> {200000}:
fastMinTxRate: -----> {0}:
interleaveMaxTxRate: -----> {200000}:
interleaveMinTxRate: -----> {0}:
rateMode: -----> {dynamic}:
maxPower: -----> {200}:
maxSnrMgn: -----> {127}:
minSnrMgn: -----> {0}:
targetSnrMgn: -----> {60}:
downshiftSnrMgn: -----> {30}:
upshiftSnrMgn: -----> {90}:
minDownshiftTime: -----> {30}:
minUpshiftTime: -----> {30}:
bitSwap: -----> {enabled}:
minINP: -----> {twosymbols}:
maxInterleaveDelay: -----> {20}:
phyRSupport: -----> {disable}: enable
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

```

zSH> update vdsl-cpe-config 1/1/1
vdsl-cpe-config 1/1/1
Please provide the following: [q]uit.
fastMaxTxRate: -----> {200000}:
fastMinTxRate: -----> {0}:
interleaveMaxTxRate: -----> {200000}:
interleaveMinTxRate: -----> {0}:
rateMode: -----> {dynamic}:
maxPower: -----> {130}:
maxSnrMgn: -----> {127}:
minSnrMgn: -----> {0}:
targetSnrMgn: -----> {60}:
downshiftSnrMgn: -----> {30}:
upshiftSnrMgn: -----> {90}:
minDownshiftTime: -----> {30}:
minUpshiftTime: -----> {30}:
bitSwap: -----> {enabled}:
minINP: -----> {twosymbols}:
maxInterleaveDelay: -----> {20}:
phyRSupport: -----> {disable}: enable
phyRmaxINP: -----> {0}:
phyRminRSoverhead: -----> {0}:
phyRRtxRatio: -----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

VDSL2 statistics

This chapter describes:

- [View VDSL2 statistics, page636](#)
- [View VDSL2 statistics with the -v variable, page637](#)
- [Clear VDSL2 counters, page638](#)
- [VDSL statistics parameters, page639](#)

View VDSL2 statistics

Use the **dslstat** *shelf/slot/port/subport/interface type* command to retrieve or clear DSL statistics for any VDSL port in the system. Entering the **dslstat** command with the -v (verbose) variable retrieves all available statistics.

Viewing VDSL2 statistics

View VDSL2 statistics.

```

zSH> dslstat 1-1-1-0/vdsl
General Stats:
-----
AdminStatus.....UP
LineStatus.....DATA
Line uptime (DD:HH:MM:SS).....0:20:34:09
DslUpLineRate (bitsPerSec).....28183000
DslDownLineRate (bitsPerSec).....86818000
DslMaxAttainableUpLineRate (bitsPerSec).....38366000
DslMaxAttainableDownLineRate (bitsPerSec)....97560000
Out Pkts/Cells.....21
Out Discards.....0
Out Errors.....0
In Pkts/Cells.....233
In Discards.....49516
In Errors.....0
DSL Physical Stats:
-----
Actual Transmission connection standard.....VDSL2
Vdsl2CurrentProfile.....g993-2-17a
DslLineSnrMgn (tenths dB).....59
DslLineAtn (tenths dB).....33
DslCurrOutputPwr (tenths dB).....145
LOFS.....0
LOLS.....0
LOSS.....0
ESS.....0
CRC Errors.....0
Inits.....0

```

View VDSL2 statistics with the -v variable

Using the -v (verbose) variable with the **dslstat** command displays all available statistics.

 **Note:** Statistics in bold indicate Phy-R™ statistics.

View all available VDSL2 statistics

To view all available statistics on VDSL2 enter:

```
zSH> dslstat 1-1-1-0/vdsl -v
General Stats:
-----
AdminStatus.....UP
LineStatus.....DATA
Line uptime (DD:HH:MM:SS).....2:00:15:23
DslUpLineRate (bitsPerSec).....38549000
DslDownLineRate (bitsPerSec).....92932000
DslMaxAttainableUpLineRate (bitsPerSec).....38513000
DslMaxAttainableDownLineRate (bitsPerSec)....94312000
Out Pkts/Cells.....6702743
Out Discards.....24846
Out Errors.....0
In Pkts/Cells.....3474987
In Discards.....220009
In Errors.....0
DSL Physical Stats:
-----
Actual Transmission connection standard.....VDSL2
Vdsl2CurrentProfile.....g993-2-17a
DslLineSnrmgn (tenths dB).....63
DslLineAtn (tenths dB).....33
DslCurrOutputPwr (tenths dB).....145
LOFS.....30
LOLS.....233
LOSS.....277
ESS.....298
CRC Errors.....21
Inits.....10
near-end statistics:
-----
Loss of Frame Seconds.....0
Loss of Signal Seconds.....277
Loss of Link Seconds.....233
Severely Errored Seconds.....277
Unavailable Seconds.....277
Retransmitted codewords.....0
Corrected Retransmitted codewords.....0
UnCorrectable Retransmitted codewords.....0
far-end statistics:
-----
```

```

Loss of Frame Seconds.....0
Loss of Signal Seconds.....247
Loss of Link Seconds.....44
Severely Errored Seconds.....247
Unavailable Seconds.....247
Retransmitted codewords.....0
Corrected Retransmitted codewords.....0
UnCorrectable Retransmitted codewords.....0
Loss of Power (dying gasps).....44
XTUC PHY Stats:
-----
serialNumber.....12l v10.03.08, 2009-11-17
vendorId.....BDCM 0x4d54
versionNumber.....VE_10_3_8
curSnrMargin (tenths dB).....63
currAtn (tenths dB).....33
currStatus.....NO DEFECT
currOutputPwr (tenths dB).....145
currAttainableRate (bitsPerSec).....94312000
currLineRate (bitsPerSec).....0
XTUC CHAN Stats:
-----
interleaveDelay (tenths milliseconds).....0
crcBlockLength (bytes).....0
currTxRate (bitsPerSec).....92932000
currTxSlowBurstProt.....0
currTxFastFec.....0
XTUR PHY Stats:
-----
serialNumber.....
vendorId.....BDCM 0
versionNumber.....A2pv6C016
curSnrMargin (tenths dB).....63
currAtn (tenths dB).....0
currStatus.....NO DEFECT
currOutputPwr (tenths dB).....76
currAttainableRate (bitsPerSec).....38513000
currLineRate (bitsPerSec).....0
XTUR CHAN Stats:
-----
interleaveDelay (tenths milliseconds).....0
crcBlockLength (bytes).....0
currTxRate (bitsPerSec).....38549000
currTxSlowBurstProt.....0
currTxFastFec.....0

```

Clear VDSL2 counters

Clearing DSL counters

You can clear DSL counters to make identifying the changing statistics easier to read.

- 1 Clear the statistics using the **dslstat clear** command

```
zSH> dslstat clear 1-1-1-0/vdsl
```

- 2 View the changes.

For reference the **dslstat** command (see [Viewing VDSL2 statistics on page 636](#)) shows the statistics prior to clearing the statistics. Statistics which are cleared by the **dslstat clear** command are in **bold**.

```
zSH> dslstat 1-1-1-0/vdsl
```

General Stats:

```
AdminStatus.....UP
LineStatus.....HANDSHAKE
DslUpLineRate (bitsPerSec).....30869000
DslDownLineRate (bitsPerSec).....85406000
DslMaxAttainableUpLineRate (bitsPerSec).....41570000
DslMaxAttainableDownLineRate (bitsPerSec)....97976000
Out Pkts/Cells.....0
Out Discards.....0
Out Errors.....0
In Pkts/Cells.....0
In Discards.....0
In Errors.....0
DSL Physical Stats:
-----
DslLineSnrMgn (tenths dB).....0
DslLineAtn (tenths dB).....0
DslCurrOutputPwr (tenths dB).....0
LOFS.....0
LOLS.....15
LOSS.....15
ESS.....15
CRC Errors.....0
Inits.....0
```

VDSL statistics parameters

[Table 70](#) defines the statistics displayed in the **dslstat** command for an VDSL line.

Table 70: VDSL2 statistics

Statistic	Description
AdminStatus	Administrative status of the port: Values: Up Interface is ready to pass packets. Down Interface is unable to pass packets. Testing Interface is in a special testing state and is unable to pass packets.

Table 70: VDSL2 statistics (Continued)

Statistic	Description
LineStatus	Line status provides information about the VDSL2 link. Values for a single VDSL2 line: ACT — the line currently has link and can pass traffic in both directions OOS — the line does not have link TRAFFIC DISABLE — The line currently has link but not underlying VDSL2 protocol; traffic will not pass.
Line uptime (DD:HH:MM:SS)	How long the interface has been up in dd hh mm (day, hour, minute, second) format.
DslUpLineRate (bitsPerSec)	Displays the DSL upstream (customer premise > central office) line rate on this interface.
DslDownLineRate (bitsPerSec)	Displays the DSL downstream (central office > customer premise) line rate on this interface.
DslMaxAttainableUpLineRate (bitsPerSec)	Displays the maximum line rate that can be supported on this line in the upstream direction.
DslMaxAttainableDownLineRate (bitsPerSec)	Displays the maximum line rate that can be supported on this line in the downstream direction.
Out Pkts/Cells	Number of transmitted packets/cells
Out Discards	Number of transmission discards.
Out Errors	Number of transmission errors.
In Pkts/Cells	Number of transmitted packets/cells
In Discards	Number of received discards.
In Errors	Number of receive errors.
VDSL2 Physical Stats:	
Actual Transmission connection standard	Indicates the maximum line rate that can be supported on this line in the downstream direction.
Vdsl2CurrentProfile	The VDSL2 standard to be used for the line.
DslLineSnrMgn (tenths dB)	DSL Line Signal to Noise Ratio Margin — The strength of the DSL signal relative to the noise on line.
DslLineAtn (tenths dB)	DSL Line Attenuation — Measure of the signal degradation between the VDSL2 port and the modem.
DslCurrOutputPwr (tenths dB)	Not currently used.
LOFS	Number of Loss of Frame Seconds.
LOLS	Number of Loss of Line Seconds.
LOSS	Number of Loss of Signal Seconds.

Table 70: VDSL2 statistics (Continued)

Statistic	Description
ESS	Number of errored seconds (the number of one-second intervals containing one or more CRC anomalies or one or more LoS or Sef defects) that has been reported in the current 15-minute interval.
CRC Errors	Cyclic Redundancy Check Errors — CRC checks for transmission errors. The CRC code is computed from the data in the message. If the data is altered the CRC computation will not be in agreement with the data.
Initis	Number of line initialization attempts, including both successful and failed attempts.
near-end (CO) statistics:	
Loss of Frame Seconds	Count of seconds during this interval that there was Loss of Framing.
Loss of Signal Seconds	Count of seconds during this interval that there was Loss of Signal.
Loss of Link Seconds	Count of seconds during this interval that there was Loss of Link.
Severely Errored Seconds	Count of Severely Errored Seconds during this interval.
Unavailable Seconds	Count of Unavailable Seconds during this interval.
Retransmitted codewords	Retransmitted Codewords.
Corrected Retransmitted codewords	Retransmitted corrected Codewords.
UnCorrectable Retransmitted codewords	Retransmitted uncorrectable Codewords.
far-end statistics:	
Loss of Frame Seconds	Count of seconds during this interval that there was Loss of Framing.
Loss of Signal Seconds	Count of seconds during this interval that there was Loss of Signal.
Loss of Link Seconds	Count of seconds during this interval that there was Loss of Link.
Severely Errored Seconds	Count of Severely Errored Seconds during this interval.
Unavailable Seconds	Count of Unavailable Seconds (UAS) during this interval.
Retransmitted codewords	Retransmitted Codewords.
Corrected Retransmitted codewords	Retransmitted corrected Codewords.
UnCorrectable Retransmitted codewords	Retransmitted uncorrectable Codewords.
Loss of Power (dying gasps)	Count of Loss of Power (LPR) Seconds during this interval.
XTUC PHY Stats:	
serialNumber	The vendor specific string that identifies the vendor equipment.
vendorID	The vendor ID code is a copy of the binary vendor identification field expressed as readable characters in hexadecimal notation.

Table 70: VDSL2 statistics (Continued)

Statistic	Description
versionNumber	The vendor specific version number sent by this Vtu as part of the initialization messages. It is a copy of the binary version number field expressed as readable characters in hexadecimal notation.
curSnrMargin (tenths dB)	Noise Margin as seen by this Vtu with respect to its received signal in 0.25dB. The effective range is -31.75 to +31.75 dB.
currAtn (tenths dB)	Measured difference in the total power transmitted by the peer Vtu and the total power received by this Vtu. The effective range is 0 to +63.75 dB.
currStatus	Indicates current state of the Vtu line. This is a bit-map of possible conditions. The various bit positions are: noDefect There are no defects on the line. lossOfFraming Vtu failure due to not receiving Frame. lossOfSignal Vtu failure due to not receiving Signal. lossOfPower Vtu failure due to loss of power. lossOfSignalQuality Loss of Signal Quality is declared when the Noise Margin falls below the Minimum Noise Margin, or the bit-error-rate exceeds 10^{-7} . lossOfLink Vtu failure due to inability to link with peer Vtu. Set whenever the transceiver is in the 'Warm Start' state. dataInitFailure Vtu failure during initialization due to bit errors corrupting. configInitFailure Vtu failure during initialization due to peer Vtu not able to support requested configuration. protocolInitFailure Vtu failure during initialization due to incompatible protocol used by the peer Vtu. noPeerVtuPresent Vtu failure during initialization due to no activation sequence detected from peer Vtu.
currOutputPwr (tenths dB)	Measured total output power transmitted by this VTU. This is the measurement that was reported during the last activation sequence.
currAttainableRate (bitsPerSec)	Indicates the maximum currently attainable data rate in steps of 1000 bits/second by the Vtu. This value will be equal to or greater than vdsIPhysCurrLineRate. Note that for SCM, the minimum and maximum data rates are equal. Note: 1 kbps = 1000 bps.
currLineRate (bitsPerSec)	Indicates the current data rate in steps of 1000 bits/second by the Vtu. This value will be less than or equal to vdsIPhysCurrAttainableRate. Note: 1 kbps = 1000 bps
XTUC CHAN Stats:	

Table 70: VDSL2 statistics (Continued)

Statistic	Description
interleaveDelay (tenths milliseconds)	Interleave Delay for this channel. Interleave delay applies only to the interleave (slow) channel and defines the mapping (relative spacing) between subsequent input bytes at the interleave input and their placement in the bit stream at the interleave output. Larger numbers provide greater separation between consecutive input bytes in the output bit stream allowing for improved impulse noise immunity at the expense of payload latency. In the case where the ifType is fast(125), return a value of zero.
crcBlockLength (bytes)	Indicates the length of the channel data-block on which the CRC operates.
currTxRate (bitsPerSec)	Actual transmit data rate on this channel. Note: 1 kbps = 1000 bps.
currTxSlowBurstProt	Actual level of impulse noise (burst) protection for an interleaved (slow) channel. This parameter is not applicable to fast channels. For fast channels, a value of zero shall be returned.
currTxFastFec	Actual Forward Error Correction (FEC) redundancy related overhead for a fast channel. This parameter is not applicable to an interleaved (slow) channel. For interleaved channels, a value of zero shall be returned.
XTUR PHY Stats:	
serialNumber	The vendor specific string that identifies the vendor equipment.
vendorId	The vendor ID code is a copy of the binary vendor identification field expressed as readable characters in hexadecimal notation.
versionNumber	The vendor specific version number sent by this Vtu as part of the initialization messages. It is a copy of the binary version number field expressed as readable characters in hexadecimal notation.
curSnrMargin (tenths dB)	Noise Margin as seen by this Vtu with respect to its received signal in 0.25dB. The effective range is -31.75 to +31.75 dB.
currAtn (tenths dB)	Measured difference in the total power transmitted by the peer Vtu and the total power received by this Vtu. The effective range is 0 to +63.75 dB.

Table 70: VDSL2 statistics (Continued)

Statistic	Description
currStatus.	Indicates current state of the Vtu line. This is a bit-map of possible conditions. The various bit positions are: noDefect There are no defects on the line. lossOfFraming Vtu failure due to not receiving. lossOfSignal Vtu failure due to not receiving. lossOfPower Vtu failure due to loss of power. lossOfSignalQuality Loss of Signal Quality is declared when the Noise Margin falls below the Minimum Noise Margin, or the bit-error-rate exceeds 10^{-7}. lossOfLink Vtu failure due to inability to link with peer Vtu. Set whenever the transceiver is in the 'Warm Start' state. dataInitFailure Vtu failure during initialization due to bit errors corrupting. configInitFailure Vtu failure during initialization due to peer Vtu not able to support requested configuration. protocolInitFailure Vtu failure during initialization due to incompatible protocol used by the peer Vtu. noPeerVtuPresent Vtu failure during initialization due to no activation sequence detected from peer Vtu.
currOutputPwr (tenths dB)	Measured total output power transmitted by this VTU. This is the measurement that was reported during the last activation sequence.
currAttainableRate (bitsPerSec)	Indicates the maximum currently attainable data rate in steps of 1000 bits/second by the Vtu. This value will be equal to or greater than vdsIPhysCurrLineRate. Note that for SCM, the minimum and maximum data rates are equal. Note: 1 kbps = 1000 bps
currLineRate (bitsPerSec)	Indicates the current data rate in steps of 1000 bits/second by the Vtu. This value will be less than or equal to vdsIPhysCurrAttainableRate. Note: 1 kbps = 1000 bps
XTUR CHAN Stats:	
interleaveDelay (tenths milliseconds)	Interleave Delay for this channel. Interleave delay applies only to the interleave (slow) channel and defines the mapping (relative spacing) between subsequent input bytes at the interleave input and their placement in the bit stream at the interleave output. Larger numbers provide greater separation between consecutive input bytes in the output bit stream allowing for improved impulse noise immunity at the expense of payload latency. In the case where the ifType is fast(125), return a value of zero.
crcBlockLength (bytes)	Indicates the length of the channel data-block on which the CRC operates.

Table 70: VDSL2 statistics (Continued)

Statistic	Description
currTxRate (bitsPerSec)	Actual transmit data rate on this channel. Note: 1 kbps = 1000 bps.
currTxSlowBurstProt	Actual level of impulse noise (burst) protection for an interleaved (slow) channel. This parameter is not applicable to fast channels. For fast channels, a value of zero shall be returned.
currTxFastFec	Actual Forward Error Correction (FEC) redundancy related overhead for a fast channel. This parameter is not applicable to an interleaved (slow) channel. For interleaved channels, a value of zero shall be returned.

VDSL2 24 port card pinouts

VDSL2 24 port cards use standard RJ-21X pinouts. [Table 71](#) lists the port pinouts.

Table 71: VDSL2 24 port card pinouts

Pin	Function	Pin	Function
1	Channel 1 ring	26	Channel 1 tip
2	Channel 2 ring	27	Channel 2 tip
3	Channel 3 ring	28	Channel 3 tip
4	Channel 4 ring	29	Channel 4 tip
5	Channel 5 ring	30	Channel 5 tip
6	Channel 6 ring	31	Channel 6 tip
7	Channel 7 ring	32	Channel 7 tip
8	Channel 8 ring	33	Channel 8 tip
9	Channel 9 ring	34	Channel 9 tip
10	Channel 10 ring	35	Channel 10 tip
11	Channel 11 ring	36	Channel 11 tip
12	Channel 12 ring	37	Channel 12 tip
13	Channel 13 ring	38	Channel 13 tip
14	Channel 14 ring	39	Channel 14 tip
15	Channel 15 ring	40	Channel 15 tip
16	Channel 16 ring	41	Channel 16 tip
17	Channel 17 ring	42	Channel 17 tip

Table 71: VDSL2 24 port card pinouts (Continued)

Pin	Function	Pin	Function
18	Channel 18 ring	43	Channel 18 tip
19	Channel 19 ring	44	Channel 19 tip
20	Channel 20 ring	45	Channel 20 tip
21	Channel 21 ring	46	Channel 21 tip
22	Channel 22 ring	47	Channel 22 tip
23	Channel 23 ring	48	Channel 23 tip
24	Channel 24 ring	49	Channel 24 tip
25	Not used	50	Not used

14 MXK EFM SHDSL CARDS

This chapter describes the MXK EFM (Ethernet in the First Mile) SHDSL cards: MXK-EFM-SHDSL-24-NTP and MXK-EFM-SHDSL-24-NTWC, G.SHDSL bonding, and configuring G.SHDSL profiles:

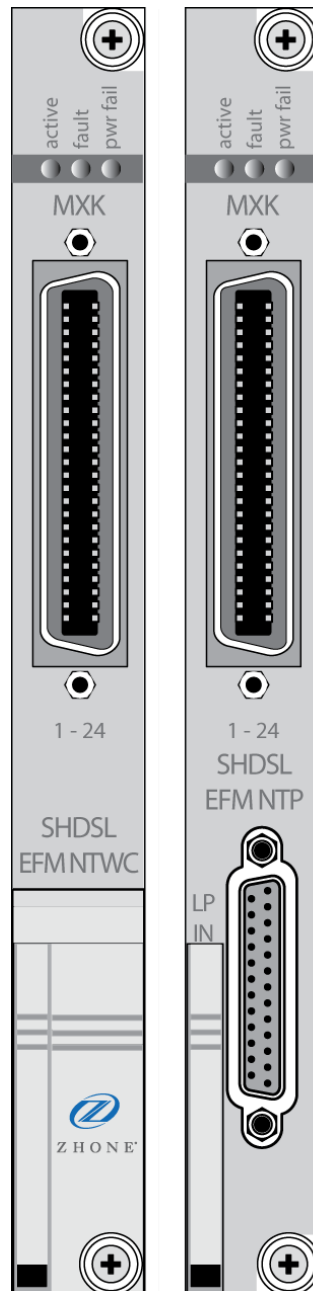
- [EFM SHDSL cards, page648](#)
- [MXK EFM SHDSL bonding overview, page653](#)
- [G. SHDSL bond group configuration, page654](#)
- [G. SHDSL bond group configuration, page654](#)
- [SNR monitoring for bonded G.SHDSL lines, page671](#)
- [SHDSL error monitoring, page664](#)
- [SHDSL statistics, page691](#)
- [Bond group statistics and port statistics, page695](#)
- [802.3ah EFM OAM, page696](#)
- [MXK-EFM-SHDSL-24 pinouts, page699](#)
- [Power and data connections for SHDSL CPE devices, page700](#)
- [MTAC testing, page703](#)

EFM SHDSL cards

This section describes the MXK EFM SHDSL 24-port cards and how to add them to the system:

- [EFM SHDSL card overview, page648](#)
- [EFM SHDSL card specifications, page649](#)
- [EFM SHDSL-24 card configuration, page650](#)

EFM SHDSL card overview



The MXK SHDSL 24-port cards provide 24 bondable SHDSL ports with a maximum of eight ports per bonded group and a maximum of 24 bonded groups.

This support enables up to three bonded groups of eight ports for 8-port EtherXtend devices, up to six bonded groups of four ports for 4-port EtherXtend devices, and up to 24 groups using one port for each EtherXtend.

The SHDSL line cards used with packet uplink cards, MXK MXK-UPLINK-2X10GE-8X1G, MXK MXK-UPLINK-8X1G, or MXK-UPLINK-4X1G, that support routing and bridging but not cell relay.

The MXK SHDSL 24-port cards provides Ethernet over SHDSL links to Zhone EtherXtends and N2N CPE devices. SHDSL links can be added or removed as the network is configured. The card automatically performs load balancing over the links.

The MXK-EFM-SHDSL-24-NTWC card provides network timing reference and current. The network timing reference allows SHDSL lines to use the backplane clock to clock T1/E1 traffic eliminating the need for a clock source at each location where remote devices are installed.



Note: The legacy Zhone SNE20x0 CPE devices present link stability issues when used with the wetting current version of the EFM-SHDSL line cards (MXK-EFM-SHDSL-24-NTWC). The issue does not exist when the SNE 20x0 are used with the Line Power, No Wetting Current version (MXK-EFM-SHDSL-24-NTP). Zhone recommends using the Line Power versions of the EFM-SHDSL cards for customers planning on deploying SNE 20x0 CPEs.

The MXK-EFM-SHDSL-24-NTP card provides network timing reference and line power. The timing reference enables the card to use the MXK timing as the SHDSL line clocking. This allows an SHDSL CPE to derive timing from the input of the SHDSL lines. It then can use that timing/clocking to provide timing to other subtended devices. The line power feature can be used to power CPEs such as the SkyZhone to eliminate the need for local power. The power is combined with the data and sent out over the 24 SHDSL ports to downstream CPE devices such as the SkyZhone. One MXK-EFM-SHDSL-24-NTP line card can provide power and data for up to 12 CPE devices.

EFM SHDSL card specifications

[Table72](#) provides the specifications for the MXK-EFM-24 bonding cards.

Table 72: MXK-EFM-SHDSL-24 card specifications

Specification	Description
Density	24 ports
Physical interface	Standard telco connector

Table 72: MXK-EFM-SHDSL-24 card specifications (Continued)

Specification	Description
Size	1 slot
Connectors	One (1) Champ 50-pin telco connector
Line characteristics	ITU G.991.2 SHDSL
Supported line rates	Symmetric rate increments up to 5.7 Mbps
Redundancy	None
Power consumption	34.0 W nominal (all port initialized, no ports trained) plus 0.79 W additional per active SHDSL interface 43.48 W maximum

EFM SHDSL-24 card configuration

This section describes how to configure the EFM SHDSL-24 card:

- [Enter a card-profile for the card, page650](#)
- [Set wetting current, page652](#)
- [Switch clocking source, page652](#)

Enter a card-profile for the card

Each card installed in the system must have a **card-profile**. The type of line card determines the parameter settings in the **card-profile** and the software image for the card. Performing a **card add** automatically creates the **card-profile** for the card with the correct software image and settings.

[Table73](#) describes the card type and software images for the MXK-EFM-SHDSL-24 cards on the MXK:

Table 73: Card type and software image

Card	Type	Name of software image
MXK-EFM-SHDSL-24-NTP	10208	<i>MXKgshdslbonded.bin</i>
MXK-EFM-SHDSL-24-NTWC	10208	<i>MXKgshdslbonded.bin</i>

Creating the card profile for EFM SHDSL-24 cards

Add a SHDSL-24 card to the system.

- 1 Install the SHDSL-24 card in the desired line card slot.
- 2 Create a **card-profile** for the card by entering:

```
zSH> card add 11
new card-profile 1/11/10208 added, sw-file-name "mxlc24gshdslbond.bin"
```

3 Verify the card by entering **slots**:

```

zSH> slots
Uplinks
  a:*MXK TWO TENGIGE EIGHT GIGE (RUNNING)
Cards
  4: MXK 4 PORT GPON (RUNNING)
 11: MXK GSHDSL-24 Bonded/with NTP (RUNNING)
 13: MXK 20 ACT ETH (RUNNING)

```

4 Verify the **card-profile** for the SHDSL-24 card:

```

zSH> get card-profile 1/11/10208
card-profile 1/11/10208
sw-file-name: -----> {mxlc24gshdslbond.bin}
admin-status: -----> {operational}
upgrade-sw-file-name: ---> {}
upgrade-vers: -----> {}
admin-status-enable: ----> {enable}
sw-upgrade-admin: -----> {reloadcurrrev}
sw-enable: -----> {true}
sw-upgrade-enable: -----> {false}
card-group-id: -----> {0}
hold-active: -----> {false}
weight: -----> {nopreference}
card-line-type: -----> {unknowntype}
card-atm-configuration: -> {notapplicable}
card-line-voltage: -----> {not-used}
maxvpi-maxvci: -----> {notapplicable}
card-init-string: -----> {}
wetting-current: -----> {disabled}
pwe-timing-mode: -----> {none}

```

5 View card information including the state of the card and how long the card has been running:

```

zSH> slots 11
Type           : MXK GSHDSL-24 Bonded
Sub-Type       : with NTP
Card Version   : 00001
EEPROM Version : 1
Serial #       : 962646
CLEI Code      : No CLEI
Card-Profile ID : 1/11/10208
Shelf          : 1
Slot           : 11
ROM Version    :
Software Version: MXK 1.16.2.109
State          : RUNNING
Mode           : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat  : 50
Fault reset    : enabled
Uptime         : 6 minutes

```

Set wetting current

The following example enables the wetting current feature. Wetting current provides 10-15 mA per G.SHDSL line. The default setting for the wetting-current parameter is disabled.



Note: The MXK G.SHDSL card cannot be pre-provisioned with wetting current. First install the card, then create the **card-profile** with **wetting-current** set to *standard*. Or pre-provision the **card-profile** without wetting current, then update the **card-profile** to enable wetting current by changing the variable to *standard* after the card is installed.



Note: Enabling wetting current from ZMS causes the card to reboot.

Setting wetting current for the MXK-EFM-SHDSL-24-NTWC card

To enable this feature, change the wetting-current parameter to standard.

```
zSH> update card-profile 1/11/10208
card-profile 1/11/10208
Please provide the following: [q]uit.
sw-file-name: -----> {mxlc24gshdslbond.bin}:
admin-status: -----> {operational}:
upgrade-sw-file-name: ---> {}:
upgrade-vers: -----> {}:
admin-status-enable: ----> {enable}:
sw-upgrade-admin: -----> {reloadcurrrev}:
sw-enable: -----> {true}:
sw-upgrade-enable: -----> {false}:
card-group-id: -----> {0}:
hold-active: -----> {false}:
weight: -----> {nopreference}:
card-line-type: -----> {unknown type}:
card-atm-configuration: -> {not applicable}:
card-line-voltage: -----> {not-used}:
maxvpi-maxvci: -----> {not applicable}:
card-init-string: -----> {}:
wetting-current: -----> {disabled} :standard
pwe-timing-mode: -----> {none}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Switch clocking source

The default clock source setting is **ntr-local-osc**. To change the default setting for the NTR setting, enter the parameter **ntr-refck-8khz**. The clock source

will switch from the local oscillator to the backplane 8Kz reference clock.
This affects all ports on the card.

Switching clocking for the MXK-EFM-SHDSL-24-NT/NTP cards

Enter **ntr-refck-8khz** in the **efmCuPmeNtr** parameter field.

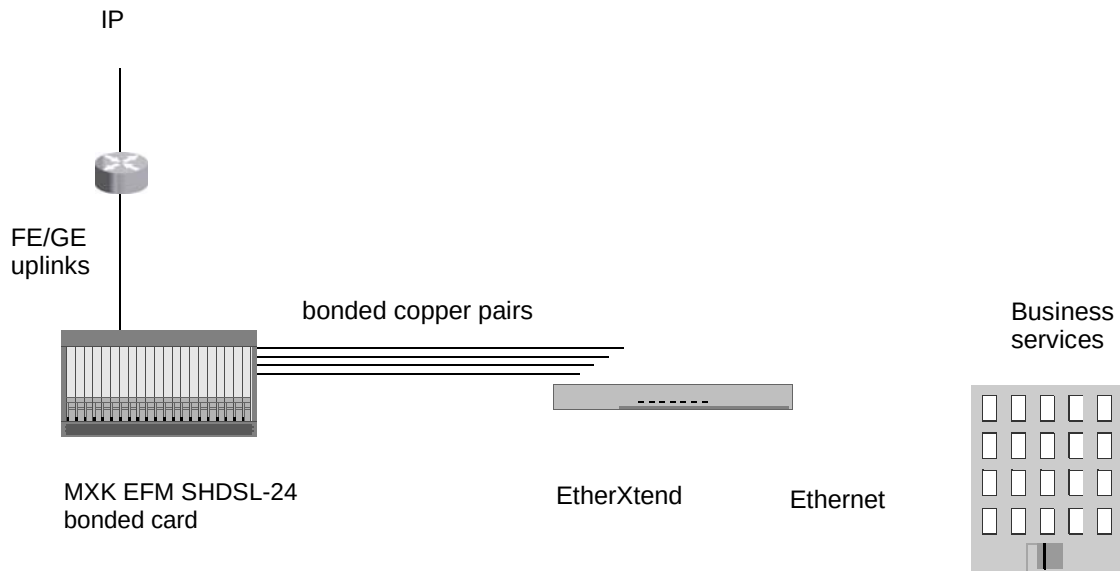
```
zSH> update pme-profile 1-11-1-0/shdsl
pme-profile 1-11-1-0/shdsl
Please provide the following: [q]uit.
efmCuPmeAdminSubType: -----> {ieee2basetlr}:
efmCuPmeAdminProfile: -----> {0}:
efmCuPAFRemoteDiscoveryCode: -----> {}:
efmCuPmeThreshLineAtn: -----> {0}:
efmCuPmeThreshMinSnrMgn: -----> {0}:
efmCuPmeLineAtnCrossingEnable: -----> {false}:
efmCuPmeSnrMgnCrossingTrapEnable: ---> {false}:
efmCuPmeDeviceFaultEnable: -----> {false}:
efmCuPmeConfigInitFailEnable: -----> {false}:
efmCuPmeProtocolInitFailEnable: -----> {false}:
efmCuPme2BProfileDescr: -----> {}:
efmCuPme2BRegion: -----> {region1}:
efmCuPme2BDataRate: -----> {0}:
efmCuPme2BPower: -----> {0}:
efmCuPme2BConstellation: -----> {adaptive}:
efmCuPme2BProfileRowStatus: -----> {active}:
efmCuPmeNtr: -----> {ntr-local-osc}: ntr-refck-8khz
efmCuPmeThreshMaxSnrMgnDelta: -----> {20}:
efmCuPmeMaintenanceMode: -----> {off}:
efmCuPmeMaintenanceStartTime: -----> {00:00}:
efmCuPmeMaintenanceEndTime: -----> {23:59}:
efmCuPmeSnrMonitoringInterval: -----> {01:00}:
efmCuPmeErrorThreshMonEnable: -----> {false}:
efmCuPmeErrorThreshMonNotifyEnable: -> {false}:
efmCuPmeErrorThreshMonInterval: -----> {12}:
efmCuPmeErrorThreshMonClrInterval: --> {1800}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

MXK EFM SHDSL bonding overview

MXK EFM (Ethernet in the first Mile) interfaces provide high speed, symmetrical Ethernet services over SHDSL links between the MXK and network extender EFM modems and Ethernet access devices like Zhone's EtherXtend and N2N CPE devices.

[Figure54](#) shows a typical network scenario using bonded copper pairs.

Figure 54: Network illustration using bonded copper pairs



G. SHDSL bond group configuration

This section describes:

- [Bond group bandwidth specifications, page655](#)
- [Bond group configuration, page655](#)
- [View bond groups, page659](#)
- [Change bond group type, page661](#)
- [Move bond group members, page662](#)
- [Delete bond groups, page662](#)
- [SHDSL error monitoring, page664](#)

MXK G.SHDSL interfaces support 802.3ah EFM standards and Zhone's proprietary Ethernet bonding technologies. MXK Ethernet bonding supports up to 24 bond groups with a maximum of eight ports in a bond group. G.SHDSL links can be added or removed from bond groups as the network is configured. The MXK automatically performs load balancing over the links.

Conditions and limitations for cross-card bonding

EFM cross-card bonding on the MXK has the following conditions and limitations:

- Cross-card bonding is not supported in N2N mode.
- Bond groups span a maximum of two cards.

- The primary card can handle a maximum of 48 ports.
- Bond groups can be created from both older and newer line cards.

Bond group bandwidth specifications

[Table74](#) shows the bond group bandwidth rates for EFM bond groups with four ports.

Table 74: Bond group bandwidth for four-port bond groups

Frame size	Downstream (pks/sec)	Upstream (pkt/sec)	Total
64	40584	40584	81168
128	21748	21748	42956
256	11105	11105	22210
512	5547	5547	11094
1024	2826	2826	5652
1280	2269	2269	4538
1480	1967	1967	3934

Bond group configuration

This section describes:

- [EFM auto bonding, page655](#)
- [EFM manual bond groups, page658](#)
- [Create bond groups on one card, page658](#)

EFM auto bonding

EFM discovery automatically groups ports that are connected to the same CPE to create a dynamic bond group utilizing automatic creation bond group numbers (201–299). The valid ranges for all EFM bond groups are:

- 25-99 for CLI created bond groups
- 101-199 for ZMS create bond groups
- 201-299 for automatic creation

The **efmCuPAFAutoDiscovery** parameter variables in the **efm-port** profile on the MXK (the CO side of the configuration) determine how EFM bond groups and their members are created.

efmCuPAFAutoDiscovery:-----> disabled optional required

[Table 75](#) defines the variables for the **efmCuPAFAutoDiscovery** parameter.

Table 75: efmCUPAFAutoDiscovery parameter

Parameter	Description
efmCuPAFAutoDiscovery	<i>disabled</i>: EFM bond groups and the members must be manually provisioned. <i>optional</i>: EFM bond groups and members may be manually or automatically provisioned. If the links were manually added to a bond group, this configuration is used and EFM Discovery is not performed. If the links were not manually added to a bond group and the CPE supports EFM Discovery, provisioning will be automatic. <i>required</i>: EFM bond groups and members are automatically provisioned with the EFM Discovery algorithm. If the CPE device does not support EFM Discovery, the link will not be obtained. Default: <i>optional</i>

EFM SHDSL cards on the MXK support up to 24 bond groups. Each bond group can have a maximum of eight members.

The number of bond groups on a SHDSL card depend on the number of ports that exist on the CPE devices connected to the EFM SHDSL card. For example, a EFM SHDSL card connected to six four-port CPE devices would have six bond groups.

Discovery works only with a discovery capable CPE device and when the EFM SHDSL **efmCuPAFAutoDiscovery** parameter is set to either *required* or *optional*.

When the parameter is set to *optional* and the CPE is not discovery capable, a dynamic bond group will not be created, but the link will remain. If the CPE is discovery capable, a dynamic bond group is created with the port as a member of the bond group. If the port already belongs to a bond group, the bond group type will be checked. If the bond group type is *dynamic*, the discovery messages are sent to verify that the ports are still connected to the same CPE. If the bond group type was created from the CLI or with ZMS, (25-99 or 101-199), discovery messages are not sent and the current configuration of bond groups remain as they are.

When the parameter is set to *required*, the CPE must be discovery capable or the links will fail. When *required* is set, discovery automatically creates bond groups depending on the number of CPE ports connected to the EFM SHDSL card.

Updating efm-port for auto-bonding

The **efmCuPAFAutoDiscovery** parameter in the **efm-port** profile must be changed to *required* or *optional* to enable auto-bonding.

Caution: The **efmCuPAFAutoDiscovery** parameter *must* have the same setting for all ports destined to the same bond group.

It is an unenforced illegal setting to have the CO **efm-port(s)** connected to the same CPE (destined to the same bond group) configured with different discovery settings.

When a configuration with different settings exists, those ports configured as *required* or *optional* will be put into one bond group, and those ports configured with disabled will either go to a different bond group or none at all causing a mismatch in traffic flow.

- 1 View the ports on the CO side that will be connected to the CPE device. The default for the **efmCuPAFAutoDiscovery** parameter is *optional* to enable *autodiscovery*.

```
zSH> get efm-port 1-4-1-0/shdsl
efm-port 1-4-1-0/shdsl
efmCuPAFAdminState: -----> {enabled}
efmCuPAFDiscoveryCode: -----> {}
efmCuAdminProfile: -----> {0x01}
efmCuTargetDataRate: -----> {50000}
efmCuTargetWorstCaseSnrMgn: -----> {0}
efmCuThreshLowBandwidth: -----> {0}
efmCuLowBandwidthEnable: -----> {false}
efmCuTargetCurrentConditionMode: ---> {false}
efmCuTargetCurrentConditionSnrMgn: -> {6}
efmCuTargetWorstCaseMode: -----> {true}
efmCuPAFAutoDiscovery: -----> {optional}
```

- 2 View the bond group automatically created.

```
zSH> bond show all
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
4	26	efmbond	ACT	1-4-26-0	-
4	25	efmbond	ACT	1-4-25-0	-
4	299	efmbond	ACT	bond-0299	-

- 3 View the links in the automatically created bond group.

```
zSH> bond show group bond-0299/efmbond
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
4	299	efmbond	ACT	bond-0299	-

Group Members					
Slot	Port	Type	State	Name	Desc
4	3	shdsl	ACT	1-4-3-0	-
4	1	shdsl	ACT	1-4-1-0	-
4	7	shdsl	ACT	1-4-7-0	-
4	2	shdsl	ACT	1-4-2-0	-
4	6	shdsl	ACT	1-4-6-0	-
4	8	shdsl	ACT	1-4-8-0	-
4	5	shdsl	ACT	1-4-5-0	-

4 4 shdsl ACT 1-4-4-0 -

EFM manual bond groups

When you are not using EFM discovery and want to create bond groups from the CLI, first create the EFM bond group, then add the links to that group before connecting the CPE.

To switch from EFM to N2N bonding, the EFM links can be moved from the bond group to a N2N bond group or disconnect the CPE and delete the EFM bond group.

The MXK SHDSL connector has 24 SHDSL ports and supports up to 24 bond groups.

Create bond groups on one card

Creating bond groups on one card

Creating a bond group requires that you first create the bond group, then add one or more bond group members.

- 1 Enter **slots** to verify the location of the EFM G.SHDSL card.

```
zSH> slots
```

```
Uplinks
```

```
a:*MXK EIGHT GIGE (RUNNING)
```

```
b: MXK EIGHT GIGE (RUNNING)
```

```
Cards
```

```
1: MXK GSHDSL-24 Bonded/with NTP (RUNNING)
```

```
2: MXK GSHDSL-24 Bonded/with NTWC (RUNNING)
```

```
4: MXK GSHDSL-24 Bonded/with NTP (RUNNING)
```

- 2 Enter **bond add group shelf-slot-port-subport/type** to create a bond group with *port* designating the group ID of the bond group.



Note: In the case of bond group commands, *port* refers to the group ID of the bond group.

When entering a bond group *interface/efmbond*, check to see which *interface* is actually created. If the bond group already exists, the system creates the interface with a system assigned value. For example,

```
zSH> bond add group 1-1-30-0/efmbond
```

```
Bond group - bond-0030/efmbond - was successfully created.
```



Note: Note that the bond group created has a different *interface* than the *interface* entered.

- 3 Verify the bond group name:

```
zSH> bond show all
```

```
Bond Groups
```

Slot	GrpId	Type	State	Name	Desc
1	30	efmbond	OOS	bond-0030	-
4	26	efmbond	ACT	1-4-26-0	-
4	25	efmbond	ACT	1-4-25-0	-
4	299	efmbond	ACT	bond-0299	-

4 Enter **bond show group interface/type** to verify the bond group.

```
zSH> bond show group bond-0030/efmbond
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	30	efmbond	OOS	bond-0030	-

5 Add a single bond group member to the bond group by entering **bond add member shelf-slot-port-subport/type shelf-slot-port-subport/type** with the interface and type of the bond group followed by the interface and type of the group member to be added.

```
zSH> bond add member bond-0030/efmbond 1-1-1-0/shdsl
```

Add several bond group members to the bond group:

```
zSH> bond add member bond-0030/efmbond 1-1-2-0/shdsl 1-1-3-0/shdsl 1-1-4-0/shdsl 1-1-5-0/shdsl 1-1-6-0/shdsl 1-1-7-0/shdsl 1-1-8-0/shdsl
```

6 Enter **bond show group shelf-slot-port-subport/type** to verify the members of the bond group.

```
zSH> bond show group bond-0030/efmbond
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	30	efmbond	OOS	bond-0030	-
Group Members					
Slot	Port	Type	State	Name	Desc
1	1	shdsl	OOS	1-1-1-0	-
1	8	shdsl	OOS	1-1-8-0	-
1	7	shdsl	OOS	1-1-7-0	-
1	6	shdsl	OOS	1-1-6-0	-
1	5	shdsl	OOS	1-1-5-0	-
1	4	shdsl	OOS	1-1-4-0	-
1	3	shdsl	OOS	1-1-3-0	-
1	2	shdsl	OOS	1-1-2-0	-

View bond groups

You can view all existing bond groups, a specific bond group, all the bond groups on a specific slot, or view the bond group of a specific link.

Viewing all bond groups

View all configured bond groups.

Enter **bond show all** to view all bond groups.

zSH> **bond show all**

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	30	efmbond	OOS	bond-0030	-
4	26	efmbond	ACT	1-4-26-0	-
4	25	efmbond	ACT	1-4-25-0	-
4	299	efmbond	ACT	bond-0299	-

Viewing a specific bond group

View a specific bond group.

Enter **bond show group shelf-slot-port-subport/type** to view the bond group and all of members of the bond group:

zSH> **bond show group bond-0030/efmbond**

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	30	efmbond	OOS	bond-0030	-
Group Members					
Slot	Port	Type	State	Name	Desc
1	1	shdsl	OOS	1-1-1-0	-
1	8	shdsl	OOS	1-1-8-0	-
1	7	shdsl	OOS	1-1-7-0	-
1	6	shdsl	OOS	1-1-6-0	-
1	5	shdsl	OOS	1-1-5-0	-
1	4	shdsl	OOS	1-1-4-0	-
1	3	shdsl	OOS	1-1-3-0	-
1	2	shdsl	OOS	1-1-2-0	-

Viewing bond groups by slot

View bond groups by slot.

Enter **bond show slot slotnumber** to view the bond groups.

zSH> **bond show slot 1**

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	30	efmbond	OOS	bond-0030	-

Viewing the bond group of a specific link

View the bond group of a specific link.

Enter **bond show link shelf-slot-port-subport/type** to view the bond group of a specific link and any other links in the group:

zSH> **bond show link 1-1-8-0/shdsl**

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	30	efmbond	OOS	bond-0030	-
Group Members					
Slot	Port	Type	State	Name	Desc
1	1	shdsl	OOS	1-1-1-0	-

1	8	shdsl	00S	1-1-8-0	-
1	7	shdsl	00S	1-1-7-0	-
1	6	shdsl	00S	1-1-6-0	-
1	5	shdsl	00S	1-1-5-0	-
1	4	shdsl	00S	1-1-4-0	-
1	3	shdsl	00S	1-1-3-0	-
1	2	shdsl	00S	1-1-2-0	-

Change bond group type

The bond group type can be changed for individual bond groups or all bond groups used in a specified slot using the **bond modify** commands.

Changing one bond group type

Change one bond group type.

- 1 Enter **bond modify type group shelf-slot-port-subport/type** to change a bond group from *efm* to *n2n*:

```
zSH> bond modify n2n group bond-0030/efmbond
```

- 2 Enter **bond show all** to view the change in type.

```
zSH> bond show all
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	30	n2nbond	00S	bond-0030	-

Changing all the bond group types on a device

To change all of the bond groups on the MXK in slot 1 from EFM to N2N enter:

- 1 View the bond group and their type.

```
zSH> bond show all
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	28	n2nbond	00S	bond-0028	-
1	27	n2nbond	00S	bond-0027	-

- 2 Enter **bond modify type slot slot number** to change the bonding type, in this case from n2n to efm.

```
zSH> bond modify efm slot 1
```

- 3 Enter **bond show all** to view the changes.

```
zSH> bond show all
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	28	efmbond	00S	bond-0028	-
1	27	efmbond	00S	bond-0027	-

Move bond group members

Bond group members can be moved from one group to another, even between bond groups of different types.

Moving bond group members

If a bond group is full, you will not be allowed to complete the move.

Enter **bond show all** to view the bond groups and their types.

```
zSH> bond show all
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	28	efmbond	00S	bond-0028	-
1	27	efmbond	00S	bond-0027	-

Enter **bond move name/type name/type shelf-slot-port-subport/type** to move a link from one group to another.

```
zSH> bond move bond-0028/efmbond bond-0027/efmbond 1-1-9-0/shdsl
```

```
zSH> bond move bond-0028/efmbond bond-0027/efmbond 1-1-10-0/shdsl
```

Caution: group bond-0028/efmbond is now empty!

The system notifies you when the bond group is empty.

Enter **bond show link shelf-slot-port-subport/type** to view the bond group where to member was moved to.

```
zSH> bond show link 1-1-9-0/shdsl
```

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	27	efmbond	00S	bond-0027	-

Group Members					
Slot	Port	Type	State	Name	Desc
1	9	shdsl	00S	1-1-9-0	-
1	10	shdsl	00S	1-1-10-0	-

Delete a member of the bond group:

Enter **bond delete member name/type shelf-slot-port-subport/type** to remove a member of a bond group.

```
zSH> bond delete member bond-0027/efmbond 1-1-9-0/shdsl
```

Delete bond groups

Bond groups can be deleted by individual member or entire group.

Deleting bond groups

Enter **bond show all** to view current bond groups, delete a bond group, and verify that the bond group is deleted.

zSH> **bond show all**

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
11	27	efmbond	00S	1-11-27-0	-
11	26	efmbond	00S	1-11-26-0	-
11	25	efmbond	00S	1-11-25-0	-

Enter **bond delete group name/type** to delete a bond group.

zSH> **bond delete group 1-11-26-0/efmbond**

Enter **bond show all** to verify the deletion.

zSH> **bond show all**

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
11	27	efmbond	00S	1-11-27-0	-
11	25	efmbond	00S	1-11-25-0	-

Cross-card bonding

Creating bond groups across cards (cross-card bonding)

Creating a bond group across cards requires that you first create the bond group, then add one or more bond group members from two cards.

- 1 Create the bond group.

zSH> **bond add group 1-1-27-0/efmbond**

Bond group - bond-0027/efmbond - was successfully created.

- 2 View the bond group.

zSH> **bond show all**

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	27	efmbond	00S	bond-0027	-

- 3 Add bond group members from two cards.

zSH> **bond add member bond-0027/efmbond 1-1-1-0/shdsl 1-1-2-0/shdsl 1-1-3-0/shdsl 1-1-4-0/shdsl 1-2-1-0/shdsl 1-2-2-0/shdsl 1-2-3-0/shdsl 1-2-4-0/shdsl**

- 4 Verify bond group members.

zSH> **bond show group bond-0027/efmbond**

Bond Groups					
Slot	GrpId	Type	State	Name	Desc
1	27	efmbond	00S	bond-0027	-

Group Members					
Slot	Port	Type	State	Name	Desc
1	1	shdsl	00S	1-1-1-0	-
2	4	shdsl	00S	1-2-4-0	-
2	3	shdsl	00S	1-2-3-0	-
2	2	shdsl	00S	1-2-2-0	-

2	1	shdsl	00S	1-2-1-0	-
1	4	shdsl	00S	1-1-4-0	-
1	3	shdsl	00S	1-1-3-0	-
1	2	shdsl	00S	1-1-2-0	-

SHDSL error monitoring

This section describes:

- [SHDSL error monitoring statistics, page664](#)
- [SHDSL error monitoring fields, page664](#)

The MXK provides the **errmon** command to view SHDSL error monitoring information.

SHDSL error monitoring statistics

Enter the **errmon stats interface/type** command to view SHDSL error monitoring statistics:

```
zSH> errmon stats 1-4-3-0/shdsl
      Shdsl Error Monitoring Stats
```

Port	TC Down	CRC	ES	SES	Max Err Sec	Restart	Line Status
3	0	0	61	61	1	0	ACT

[Table76](#) defines the **errmon stats** fields.

Table 76: Definitions of displayed statistics

Parameter	Description
TC Down	Count of how many times the TC layer went down since the physical link was obtained.
CRC	Count of CRC anomalies.
ES	Count of one second intervals during which one or more CRCs are reported.
SES	Count of one second intervals during which at least 50 CRCs are reported.
Max Errored Sec	Maximum consecutive seconds with errors without causing action to be taken by errmon features.
Restart	Count of the number of times the port was restarted by errmon features.
Line Status	Current status of the port.

SHDSL error monitoring fields

The **errmon interface/type** command monitors the physical interface, for example, *1-5-1-0/shdsl*.

Enter `errmon show` to view the SHDSL error monitoring fields:

```
zSH> errmon show 1-4-3-0/shdsl
                Shdsl Error Monitoring
Port  Monitor  Notify  ErrorInt  ClearInt  Line Status
  3    disabled disabled      12      1800      ACT
```

The *Monitor* and the *Notify* fields must be enabled from the CLI:

```
zSH> errmon modify 1-4-3-0/shdsl monitor enable errmon show
1-4-3-0/shdsl
                Shdsl Error Monitoring
Port  Monitor  Notify  ErrorInt  ClearInt  Line Status
  3    enabled disabled      12      1800      ACT

zSH> errmon modify 1-4-3-0/shdsl notify enable errmon show 1-4-3-0/shdsl
                Shdsl Error Monitoring
Port  Monitor  Notify  ErrorInt  ClearInt  Line Status
  3    enabled  enabled      12      1800      ACT
```

Table 77: errmon parameters

Parameter	Description
link interface (port)	Name and type of a physical interface. For example, <i>1-5-1-0/shdsl</i> .
monitor	Determines if error monitoring should be performed on the link.
notify	Determines if a notification to the CLI, alarm manager, and ZMS should be generated when an error threshold is exceeded or cleared on the link.
errinterval	Specifies the number of consecutive seconds of detecting errors that, once reached, causes the physical line to be considered a poor performer and action to be taken.
clrinterval	Specifies the number of consecutive error-free seconds that must be achieved in order to declare the physical line usable for transporting data traffic.

errmon modify

Syntax `error modify|show|stats link/interface`

Configure the pme-profile

A **pme-profile** (Physical Medium Entities) is available for each G.SHDSL port. PME profiles are used to set link rates.

To display the PME parameters in their default state, enter **get pme-profile shelf-slot-port-subport/type**:

```
zSH> get pme-profile 1-1-1-0/shdsl
```

```

pme-profile 1-1-1-0/shdsl
efmCuPmeAdminSubType: -----> {ieee2basetlr}
efmCuPmeAdminProfile: -----> {0}
efmCuPAFRemoteDiscoveryCode: -----> {}
efmCuPmeThreshLineAtn: -----> {0}
efmCuPmeThreshMinSnrMgn: -----> {0}
efmCuPmeLineAtnCrossingEnable: -----> {false}
efmCuPmeSnrMgnCrossingTrapEnable: ---> {false}
efmCuPmeDeviceFaultEnable: -----> {false}
efmCuPmeConfigInitFailEnable: -----> {false}
efmCuPmeProtocolInitFailEnable: -----> {false}
efmCuPme2BProfileDescr: -----> {}
efmCuPme2BRegion: -----> {region1}
efmCuPme2BDataRate: -----> {0}
efmCuPme2BPower: -----> {0}
efmCuPme2BConstellation: -----> {adaptive}
efmCuPme2BProfileRowStatus: -----> {active}
efmCuPmeNtr: -----> {ntr-local-osc}
efmCuPmeThreshMaxSnrMgnDelta: -----> {20}
efmCuPmeMaintenanceMode: -----> {off}
efmCuPmeMaintenanceStartTime: -----> {00:00}
efmCuPmeMaintenanceEndTime: -----> {23:59}
efmCuPmeSnrMonitoringInterval: -----> {01:00}
efmCuPmeErrorThreshMonEnable: -----> {false}
efmCuPmeErrorThreshMonNotifyEnable: -> {false}
efmCuPmeErrorThreshMonInterval: -----> {12}
efmCuPmeErrorThreshMonClrInterval: --> {1800}

```

Configure automatic baud rate adaption and fixed rate settings

The line type for an G.SHDSL interface on the MXK is set at **shdsl-2btl** which can perform automatic baud rate adaption. This allows receiving devices to communicate with transmitting devices operating at different baud rates without the need to establish data rates in advance. By determining the baud rate from the transmitting device, the receiving MXK automatically trains to match the line rate of the incoming data.

The automatic baud rate adaption process may take several minutes. This is because the MXK (co) and CPE devices use an algorithm to step through a sequence of baud rates, where the devices establish a connection at each line rate and then move to the next higher rate until they reach the final rate they agree upon.

For the **efmCuPme2BDataRate** parameter, setting the parameter to 0 sets the data rate to adaptive. Entering a value defines a specific data rate in kbps.

```
efmCuPme2BDataRate:-----> {0 - 15352}
```

Before adjusting the data rate in the **efmCuPmeDataRate** parameter, see [Table 78](#) to understand the implications of setting data rates for both co and

cpe mode. [Table79](#) describes the adaptive [fixed-rate=0] and fixed line rate settings defined in the **efmCuPme2BDataRate** entry of the **pme-profile**.

Table 78: Fix-bit-rate settings and modem train rates

CO	CPE	Then
efmCuPme2BDataRate = 0	efmCuPme2BDataRate = 0	Highest available rate is negotiated.
efmCuPme2BDataRate = 0	efmCuPme2BDataRate = non-zero value = x	x is treated as the MAXIMUM train rate allowed. Modems train at x or less.
efmCuPme2BDataRate = non-zero value = x	efmCuPme2BDataRate = 0	x is treated as the MAXIMUM train rate allowed. Modems train at x or less.
efmCuPme2BDataRate = non-zero value = x	efmCuPme2BDataRate = non-zero-value = x	x is treated as the MAXIMUM train rate allowed. Modems train at x or less.
efmCuPme2BDataRate = non-zero value = x	efmCuPme2BDataRate = non-zero-value = y	The lesser of x and y = z will be treated as the MAXIMUM train rate allowed. Modems train at z or less.

Configure auto-negotiate or specific data rate

For the **efmCuPme2BDataRate** parameter, setting the parameter to 0 sets the data rate to *adaptive*. Entering a data rate value defines a specific data rate that must correspond with the constellation set in the **efmCuPme2BConstellation** parameter (see [Table79](#)).

For TCPAM16 and TCPAM32 you can set the **efmCuPme2BDataRate** either to adaptive (0) or a data rate corresponding to the TCPAM setting.



Note: When TCPAM16 and TCPAM32 are set to adaptive (0), the maximum data rate is always 5696 kbps on each line configured as adaptive.

For TCPAM4, TCPAM8, and TCPAM64, a data rate *must* be set in the **efmCuPme2BDataRate** parameter. The adaptive rate of 0 (auto-negotiate) cannot be used for these constellation settings. See [Table79](#) for data rate ranges.



Note: Configuring TCPAM is necessary only on the CO side. However, the CPE must be capable of extended rates in order for the line to successfully train/link.

TCPAM settings are typically used as follows:

- TCPAM4 and TCPAM8 are useful for operations over longer and/or noisier loops, but use reduced data rates.
- TCPAM64 is useful on short loops using CAT5 wiring for in-building and campus applications. In these cases, CAT3 loops would be too noisy for the SNR requirements.

 **Note:** When configuring the MXK for TCPAM 64, the CAT5 rating needs to apply to all intermediary connection points and accessories such as:

- 66-blocks/110-blocks
- jumper wires used on cross-connect blocks, protector blocks, etc.

It is also recommended that the number of RJ21/AMP connectors used between the DSLAM and the final cable are kept to a minimum. It is also recommended that you use CAT5 punch-down blocks as opposed to those blocks that are pre-terminated with AMP connectors.

Configure constellation for a TCPAM setting

TCPAM configurations have a minimum and a maximum data rate range, depending on the constellation setting. If a data rate is set outside of the correct range of the constellation setting, an error message will return.

[Table79](#) provides the settings for the **efmCuPme2BConstellation** parameter and the rate range for the **efmCuPme2BDataRate** parameter.

When the train rate is set to *adaptive* mode, the specific value configured in the **efmCuPme2BDataRate** field becomes the maximum rate. This means that the line will train successfully to any train rate at or below the configured value and will train at the specified rate unless line conditions dictate otherwise.

When the constellation is not *adaptive* (i.e. is set to *tcpam4*, *tcpam8*, *tcpam16*, *tcpam32*, or *tcpam64*), the specific value set in the **efmCuPme2BDataRate** field becomes fixed. This means that if the requested data rate cannot be achieved due to line conditions, the line will not get link.

 **Note:** N2N bonding supports TCPAM16/32, but not TCPAM4/8/64.

Table 79: efmCuPme2BConstellation and efmCuPme2BDataRate parameter settings

efmCuPme2BConstellation settings	efmCuPme2BData range rates
TCPAM4	192 to 2048
TCPAM8	192 to 5056
TCPAM16	192 to 7616 or an adaptive rate of 0 with a max train rate of 5696

Table 79: efmCuPme2BConstellation and efmCuPme2BDataRate parameter settings (Continued)

efmCuPme2BConstellation settings	efmCuPme2BData range rates
TCPAM32	768 to 10176 or an adaptive rate of 0 with a max train rate of 5696
TCPAM64	768 to 12736
Adaptive	0 or 192 to 5696

Configuring the constellation for a TCPAM setting

Update the **pme-profile** for TCPAM and data rates.

This example sets the **efmCuPme2BDataRate** parameter to 12736 and changes the **efmCuPme2BConstellation** parameter to *tcpam64*.

```

zSH> update pme-profile 1/1/1/0/shdsl
pme-profile 1/1/1/0/shdsl
Please provide the following: [q]uit.
efmCuPmeAdminSubType: -----> {ieee2basetlr}:
efmCuPmeAdminProfile: -----> {0}:
efmCuPAFRemoteDiscoveryCode: -----> {}:
efmCuPmeThreshLineAtn: -----> {0}:
efmCuPmeThreshMinSnrMgn: -----> {0}:
efmCuPmeLineAtnCrossingEnable: -----> {false}:
efmCuPmeSnrMgnCrossingTrapEnable: ---> {false}:
efmCuPmeDeviceFaultEnable: -----> {false}:
efmCuPmeConfigInitFailEnable: -----> {false}:
efmCuPmeProtocolInitFailEnable: -----> {false}:
efmCuPme2BProfileDescr: -----> {}:
efmCuPme2BRegion: -----> {region1}:
efmCuPme2BDataRate: -----> {0}: 12736
efmCuPme2BPower: -----> {0}:
efmCuPme2BConstellation: -----> {adaptive}: tcpam64
efmCuPme2BProfileRowStatus: -----> {active}:
efmCuPmeNtr: -----> {ntr-local-osc}:
efmCuPmeThreshMaxSnrMgnDelta: -----> {20}:
efmCuPmeMaintenanceMode: -----> {off}:
efmCuPmeMaintenanceStartTime: -----> {00:00}:
efmCuPmeMaintenanceEndTime: -----> {23:59}:
efmCuPmeSnrMonitoringInterval: -----> {01:00}:
efmCuPmeErrorThreshMonEnable: -----> {false}:
efmCuPmeErrorThreshMonNotifyEnable: -> {false}:
efmCuPmeErrorThreshMonInterval: -----> {12}:
efmCuPmeErrorThreshMonClrInterval: --> {1800}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

Here is the error message that is returned when the data rate is set too high for the constellation type.

```

Pme Data Rate and Constellation values are not compatible
tcpam32 has a Max Data Rate of 10176

```

Here is the error message that is returned when the data rate is set too low for the constellation type.

Pme Data Rate value invalid for this device

Here is the error message if you try to set an adaptive rate for TCPAM that does not support the adaptive rate:

```
zSH> update pme-profile 1/1/1/0/shdsl
pme-profile 1/1/1/0/shdsl
Please provide the following: [q]uit.
efmCuPmeAdminSubType: -----> {ieee2basetlr}:
efmCuPmeAdminProfile: -----> {0}:
efmCuPAFRemoteDiscoveryCode: -----> {}:
efmCuPmeThreshLineAtn: -----> {0}:
efmCuPmeThreshMinSnrMgn: -----> {0}:
efmCuPmeLineAtnCrossingEnable: -----> {false}:
efmCuPmeSnrMgnCrossingTrapEnable: ---> {false}:
efmCuPmeDeviceFaultEnable: -----> {false}:
efmCuPmeConfigInitFailEnable: -----> {false}:
efmCuPmeProtocolInitFailEnable: -----> {false}:
efmCuPme2BProfileDescr: -----> {}:
efmCuPme2BRegion: -----> {region1}:
efmCuPme2BDataRate: -----> {14912}: 0
efmCuPme2BPower: -----> {0}:
efmCuPme2BConstellation: -----> {tcpam4}: tcpam64
efmCuPme2BProfileRowStatus: -----> {active}:
efmCuPmeNtr: -----> {ntr-local-osc}:
efmCuPmeThreshMaxSnrMgnDelta: -----> {20}:
efmCuPmeMaintenanceMode: -----> {off}:
efmCuPmeMaintenanceStartTime: -----> {00:00}:
efmCuPmeMaintenanceEndTime: -----> {23:59}:
efmCuPmeSnrMonitoringInterval: -----> {1:00}:
efmCuPmeErrorThreshMonEnable: -----> {false}:
efmCuPmeErrorThreshMonNotifyEnable: -> {false}:
efmCuPmeErrorThreshMonInterval: -----> {12}:
efmCuPmeErrorThreshMonClrInterval: --> {1800}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Pme Data Rate and Constellation values are not compatible
tcpam64 has a Data Rate range of [768 - 12736]
Starting over....
efmCuPmeAdminSubType: -----> {ieee2basetlr}:
```

Set a region

For the **efmCuPme2BRegion** parameter, the regions are set as specified in the relevant Regional Annex of [G.991.2]. Regional settings place limitation on the maximum allowed data rate, power, and constellation. The possible values for this parameter are:

- Region 1
Annex A and F (North America)

- Region 2
Annex B and G (Europe)

You can only change regions when the link is down.

SNR monitoring for bonded G.SHDSL lines

This section describes how the MXK provides SNR monitoring for G.SHDSL lines:

- [SNR monitoring for the MXK, page671](#)
- [MXK SNR monitoring pme-profile parameters, page673](#)
- [Usage for SNR pme-profile and efm-port parameters, page674](#)
- [MXK SNR monitoring configuration, page675](#)
- [Verify SNR monitoring is enabled/disabled, page683](#)
- [G. SHDSL SNR monitoring example, page684](#)
- [Disable SNR monitoring, page689](#)

SNR monitoring for the MXK

This section describes how SNR monitoring for the MXK works:

- [SNR monitoring for the MXK overview, page671](#)
- [Current condition SNR maximum threshold, page672](#)
- [Current condition minimum SNR threshold, page672](#)

SNR monitoring for the MXK overview

Seamless Rate Adaptation (SRA) is found on ADSL2 lines to monitor the current Signal to Noise Ratio (SNR) and seamlessly adjusts the rate of the line to keep the current SNR at the target SNR. SRA prevents potential errors when the SNR drops by sacrificing train rate and allows for a higher train rate if the SNR improves. Because SHDSL lines **do not** have a comparable feature, when SHDSL forces a line to retrain to get a better rate, seamless retraining is not possible because traffic does not flow on links that are retraining.

For MXK G.SHDSL bonded lines, it is possible to configure SNR monitoring to monitor the current SNR and retrain the link when the SNR rises above or drops below target thresholds for current condition only. When retraining is performed on SHDSL bond groups of two or more ports, one link per bond group retrains at a time causing the performance of the bond group to become somewhat degraded but does not eliminate traffic flow entirely. Retraining on bond groups of just one port causes the bond group to go down. The system performs the retraining of G.SHDSL links only during a maintenance period

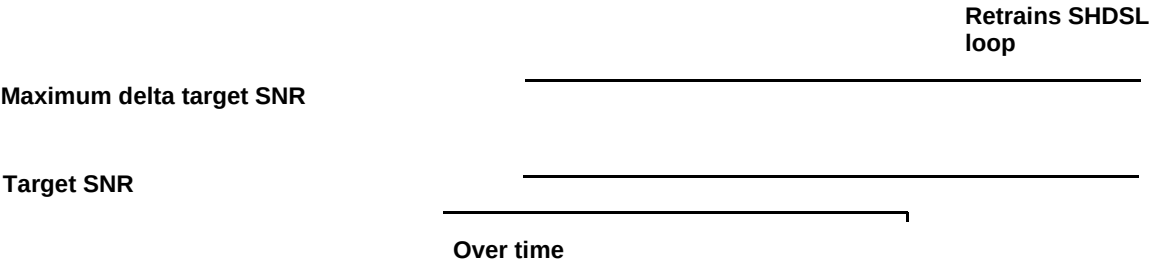
that you must specify. This maintenance period can be an entire twenty four hour day or any portion of a twenty four hour day. After a link retrain, the MXK waits a maximum of 3 minutes for the link to come up before retraining the next port.

To configure SNR monitoring, a target SNR is set in the **efm-port** profile for current condition or worst case SNR. Additionally, a maximum delta SNR threshold and a minimum SNR threshold is set in the **pme-profile**. SNR monitoring is configured for either current condition or for worst case SNR.

Current condition SNR maximum threshold

Current condition SNR determines the maximum threshold that causes the link to retrain by checking two SNR settings, the **efmCuTargetCurrentConditionSnrMgn** value in the **efm-port** profile and the **efmCuPmeThresMaxSnrMgnDelta** value in the **pme-profile**. When the current SNR exceeds the sum of these two values, the system retrains the link during the maintenance period.

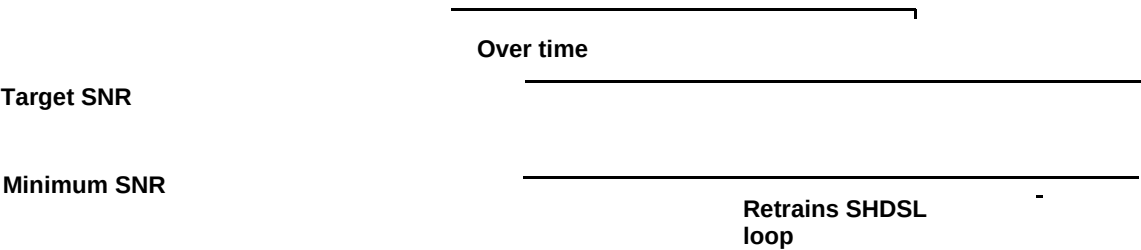
Figure 55: SNR monitoring for current condition maximum threshold



Current condition minimum SNR threshold

Current condition SNR determines the minimum threshold to cause the link to retrain by checking the **efmCuPmeThreshMinSnrMgn** value in the **pme-profile**. When the current SNR drops below this value, the system retrains the link during the maintenance period.

Figure 56: SNR monitoring for current condition minimum threshold



MXK SNR monitoring pme-profile parameters

The following parameters are used to configure the MXK for SNR monitoring and are set in the **pme-profile**:

- **efmCuPmeMaintenanceMode**

The variable settings of this parameter determine the current maintenance mode operational status. [Table80](#) describes the **efmCuPmeMaintenanceMode** variables and their functions.

Table 80: SNR modes

Variable	Function
<i>off</i> (default)	The <i>off</i> setting indicates there is no SNR monitoring.
<i>manual</i>	The <i>manual</i> setting retrains the link one time only regardless of the SNR setting during the maintenance period set in the efmCuPmeMaintenanceStartTime and the efmCuPmeMaintenanceEndTime parameters.
<i>automatic-once</i>	The <i>automatic-once</i> setting retrains the link one time only during the maintenance period set in the efmCuPmeMaintenanceStartTime and the efmCuPmeMaintenanceEndTime parameters when the SNR value is outside of the specified threshold.
<i>automatic-daily</i>	The <i>automatic-daily</i> setting retrains the link each day during the maintenance period set in the efmCuPmeMaintenanceStartTime and the efmCuPmeMaintenanceEndTime parameters when the SNR value is outside of the specified threshold.
<i>automatic-continuous</i>	The <i>automatic-continuous</i> setting retrains the link according to the start and end times in the efmCuPmeMaintenanceStartTime and the efmCuPmeMaintenanceEndTime parameters at the interval set in the efmCuPmeSnrMonitoringInterval parameter when the SNR value is outside of the specified threshold.

- **efmCuPmeMaintenanceStartTime**

This parameter provides the start time for maintenance to retrain the link in *manual* maintenance mode. When maintenance mode is set to *automatic once* or *automatic daily*, this parameter provides the start time for monitoring of the SNR value that considers the SNR threshold values specified in **efmCuPmeThreshMinSnrMgn** and **efmCuPmeThreshMaxSnrMgnDelta** in the **pme-profile**.

The start time format is HH:MM where HH is the military time for hour (0-23) and MM is the military time for minutes (0-59).

- **efmCuPmeMaintenanceEndTime**

This parameter provides the end time for maintenance to retrain the link in *manual* maintenance mode. When maintenance mode is set to *automatic once* or *automatic daily*, this parameter provides the end time for monitoring of the SNR value that considers the SNR threshold values specified in **efmCuPmeThreshMinSnrMgn** and **efmCuPmeThreshMaxSnrMgnDelta** in the **pme-profile**.

The end time format is HH:MM where HH is the military time for hour (0-23) and MM is the military time for minutes (0-59).

- **efmCuPmeSnrMonitoringInterval**

This parameter provides the SNR monitoring interval for how frequently monitoring of the SNR occurs depending on the SNR threshold values specified in **efmCuPmeThreshMinSnrMgn** and **efmCuPmeThreshMaxSnrMgnDelta**.

The interval time format is HH:MM where HH is the military time for hour (0-23) and MM is the military time for minutes (0-59).

Usage for SNR pme-profile and efm-port parameters



Note: When an EFM link is configured for a fixed rate in the **pme-profile** member **efmCuPme2BDataRate** (a value other than 0) the SNR monitoring feature should not be used.

See [Configure automatic baud rate adaption and fixed rate settings on page666](#), [Verify SNR monitoring is enabled/disabled on page683](#) and [Disable SNR monitoring on page689](#).

For SNR monitoring, the maintenance mode settings **automatic-once**, **automatic-daily**, and **automatic-continuous** require the system to check for either the minimum or the maximum threshold settings. The minimum SNR threshold is set in the **efmCuPmeThreshMinSnrMgn** parameter in the **pme-profile**. The maximum SNR threshold is the sum of the **efmCuPmeThreshMaxSnrMgnDelta** in the **pme-profile** and the **efmCuTargetWorstCaseSnrMgn** or the **efmCuTargetCurrentConditionSnrMgn** in the **efm-port** profile. The target mode, either worst case or current condition, determines which variable settings are checked. These modes are set in the **efm-port** profile parameters **efmCuTargetCurrentConditionMode** or **efmCuTargetWorstCaseMode**.

- **efmCuPmeThreshMinSnrMgn**

For both current condition mode and worst case mode, forces the link to retrain to improve the SNR when the SNR falls below this setting. The default **efmCuPmeThreshMinSnrMgn** is 0 dB.

The link retrains during the maintenance window when the current **DslLineSnrMgn** is less than the **efmCuPmeThreshMinSnrMgn** setting.

Use the **dslstat** command to view the **DslLineSnrMgn** (in tenths dB).

$$(\text{DslLineSnrMgn}/10) < \text{efmCuPmeThreshMinSnrMgn}$$

- **efmCuPmeThreshMaxSnrMgnDelta**, **efmCuTargetWorstCaseSnrMgn**, or **efmCuTargetCurrentConditionSnrMgn**

For both worst case mode and current condition mode, forces the link to retrain to improve the SNR when the SNR rate rises above the sum of the **efmCuPmeThreshMaxSnrMgnDelta** and the **efmCuTargetWorstCaseSnrMgn** or the **efmCuTargetCurrentConditionSnrMgn**.

For worst case mode, use the **dslstat** command to view the **DslLineSnrMgn** (in tenths dB).

$(\text{DslLineSnrMgn}/10) > (\text{efmCuTargetWorstCaseSnrMgn} + \text{efmCuPmeThreshMaxSnrMgnDelta})$

For current condition mode, use the **dslstat** command to view the **DslLineSnrMgn** (in tenths dB).

$(\text{DslLineSnrMgn}/10) > (\text{efmCuTargetCurrentConditionSnrMgn} + \text{efmCuPmeThreshMaxSnrMgnDelta})$

MXK SNR monitoring configuration

This section provides MXK SNR configuration procedures:

- [Set SNR for target current condition or target worst case mode, page675](#)
- [Set MXK time and day, page676](#)
- [Set SNR monitoring from the CLI, page676](#)
- [View SNR monitoring statistics, page679](#)
- [Set SNR monitoring in the pme-profile, page680](#)
- [Configure SNR crossing traps, page683](#)



Note: If the maintenance mode, **minsnrnmgn**, **maxdeltasnrnmgn**, or the monitoring interval setting is changed, then the monitoring of the SNR values starts over again after 30 seconds.

Set SNR for target current condition or target worst case mode

SNR monitoring is set for either current condition mode or worst case mode. These modes are set to *true* or *false* in the **efm-port** profile parameters **efmCuTargetCurrentConditionMode** or **efmCuTargetWorstCaseMode**.

The target SNR value for current condition and worst case mode is set in the **efmCuTargetWorseCaseSnrMgn** (default 0db) and **efmCuTargetCurrentConditionSnrMgn** (default 6db) parameters.

Setting SNR for target current condition or target worst case mode

To set SNR monitoring to monitor for worst case SNR enter:

```
zSH> update efm-port 1-11-1-0/shdsl
efm-port 1-11-1-0/shdsl
Please provide the following: [q]uit.
efmCuPAFAdminState: -----> {enabled}:
efmCuPAFDiscoveryCode: -----> {}:
efmCuAdminProfile: -----> {0x01}:
efmCuTargetDataRate: -----> {50000}:
efmCuTargetWorstCaseSnrMgn: -----> {0}:
efmCuThreshLowBandwidth: -----> {0}:
efmCuLowBandwidthEnable: -----> {false}:
efmCuTargetCurrentConditionMode: ---> {false}: false
efmCuTargetCurrentConditionSnrMgn: -> {6}:
efmCuTargetWorstCaseMode: -----> {false}: true
efmCuPAFAutoDiscovery: -----> {disabled}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Set MXK time and day

Set the MXK time and day if not already set.

Setting the MXK time and day

You can either use the **setdatettime** command or use SNTP (the network time protocol) to provide time to the MXK rather than set it manually. For example:

```
zSH> setdatettime < month(mm) day(dd) year(yyyy) hour(hh) minute(mm) second(ss) >
```

Set SNR monitoring from the CLI

Use the **snrmon** command to enable SNR monitoring by setting minimum and maximum SNR margins for the link, and the administrative maintenance mode, intervals, and time for the link.

```
snrmon <modify|show|stats> <link interface>
```

Table 81: snrmon command

Variable	Function
link interface	Name and type of a physical interface. (i.e. 1-2-1-0/shdsl).
minsnr (minimum SNR margin)	Desired minimum snr margin threshold for the link.
deltasnr (maximum delta SNR margin)	Desired maximum snr margin threshold delta for the link.

Table 81: snrmon command (Continued)

Variable	Function
mode	Desired administrative maintenance mode of the link. off manual automatic-once automatic-daily automatic-continuous
trap	Determines if traps should be generated for threshold crossings of the link. enable disable
interval	Actual frequency that snr monitoring should occur for this link. HH:MM
start	Actual time of day (in 24 hour notation) that snr monitoring should start for this link. HH:MM
end	Actual time of day (in 24 hour notation) that snr monitoring should end for this link. HH:MM
Displayed statistics:	
Snr	Obtained SNR (in tenths DB) value for the physical line.
Snr Mgn Crossing Cnt	Count of each hour when SNR mgn is out of range.
Snr Min	Configured Min SNR value.
Snr Delta	Configured Delta SNR value.
Restart	Count of the number of times the port was restarted by snrmon feature.
Line Status	Current status of the port.

Setting SNR monitoring from the CLI

1 View the SHDSL SNR monitoring fields:

```

zSH> snrmon show 1-7-1-0/shdsl
                Shdsl Snr Monitoring
Port  Snr Min Snr Delta  Start    End    Interval  Trap    Mode
1      0      20      00:00   23:59   01:00   disabled  off

```

2 Set the desired minimum SNR margin for the link and verify the setting:

```

zSH> snrmon modify 1-7-1-0/shdsl minsnr 2

zSH> snrmon show 1-7-1-0/shdsl

```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	2		20		00:00	23:59	01:00	disabled	off

The **efmCuPmeThreshMinSnrMgn** parameter in the **pme-profile** is now 2.

```
efmCuPmeThreshMinSnrMgn: -----> {2}
```

3 Set the desired maximum SNR margin delta and verify the setting:

```
zSH> snrmon modify 1-7-1-0/shdsl deltasnr 16
```

```
zSH> snrmon show 1-7-1-0/shdsl
```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	2		16		00:00	23:59	01:00	disabled	off

The **efmCuPmeThreshMaxSnrMgnDelta** parameter in the **pme-profile** is now set to 16.

```
efmCuPmeThreshMaxSnrMgnDelta: -----> {16}
```

4 Set the start of the maintenance period and verify the setting:

```
zSH> snrmon modify 1-7-1-0/shdsl start 01:00
```

```
zSH> s nrmonshow 1-7-1-0/shdsl
```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	2		16		01:00	23:59	01:00	disabled	off

The **efmCuPmeMaintenanceStartTime** parameter in the **pme-profile** is now set to 1:00.

```
efmCuPmeMaintenanceStartTime: -----> {1:00}
```

5 Set the end of the maintenance period and verify the setting:

```
zSH> snrmon modify 1-7-1-0/shdsl end 02:00
```

```
zSH> snrmon show 1-7-1-0/shdsl
```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	2		16		01:00	02:00	01:00	disabled	off

The **efmCuPmeMaintenanceEndTime** parameter in the **pme-profile** is now set to 2:00.

```
efmCuPmeMaintenanceEndTime: -----> {2:00}
```

6 Set the maintenance monitoring interval and verify the setting:

```
zSH> snrmon modify 1-7-1-0/shdsl interval 00:20
```

```
zSH> snrmon show 1-7-1-0/shdsl
```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	2		16		01:00	02:00	00:20	disabled	off

The **efmCuPmeSnrMonitoringInterval** in the **pme-profile** is now set to *00:20*.

```
efmCuPmeSnrMonitoringInterval: -----> {00:20}
```

7 Enable the trap setting and verify the setting:

```
zSH> snrmon modify 1-7-1-0/shdsl trap enable
```

```
SH> snrmon show 1-7-1-0/shdsl
```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	2		16		01:00	02:00	00:20	enabled	off

The **efmCuPmeSnrMgnCrossingTrapEnable** parameter is now set to *true*:

```
efmCuPmeSnrMgnCrossingTrapEnable: ---> {true}
```

8 Set the SNR maintenance mode and verify the setting:

```
zSH> snrmon modify 1-7-1-0/shdsl mode automatic-daily
```

```
zSH> snrmon show 1-7-1-0/shdsl
```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	2		16		01:00	02:00	00:20	disabled	automatic-daily

The **efmCuPmeMaintenanceMode** parameter is now set to **automatic-daily**.

```
efmCuPmeMaintenanceMode: -----> {automatic-daily}
```

View SNR monitoring statistics

Viewing SNR monitoring statistics

Enter **snrmon stats** to view SNR error monitoring statistics:

```
zSH> snrmon stats 1-7-1-0/shdsl
```

Shdsl Snr Monitoring Stats	
Snr	Snr

Port	(in tenths db)	Crossing Cnt	Snr Min	Snr Delta	Restart	Line Status
1	180	0	0	20	0	ACT

Set SNR monitoring in the pme-profile

Zhone recommends using the CLI macro commands to configure SNR monitoring. Setting SNR monitoring in the **pme-profile** is for advanced system configuration users.

To set the maintenance mode for the MXK, enter the value for the **efmCuPmeMaintenanceMode** parameter of the **pme-profile**:

To set the maintenance period when SNR monitoring occurs, enter a start time and an end time for the **efmCuPmeMaintenanceStartTime** and the **efmCuPmeMaintenanceEndTime** parameters.

Setting the maintenance mode and time period in the pme-profile

- 1 View the current time on the MXK, if necessary, with **showdatetime**:

```
zSH> showdatetime
Current Time: TUE FEB 09 09:49:52 2010
```

- 2 For *automatic-once* mode, set the **efmCuPmeMaintenanceMode** to *automatic-once* and set the **efmCuPmeMaintenanceStartTime** and the **efmCuPmeMaintenanceEndTime** for the time of the maintenance period.

```
zSH> update pme-profile 1-5-1-0/shdsl
pme-profile 1-5-1-0/shdsl
Please provide the following: [q]uit.
efmCuPmeAdminSubType: -----> {ieee2basetlr}:
efmCuPmeAdminProfile: -----> {0}:
efmCuPAFRemoteDiscoveryCode: -----> {}:
efmCuPmeThreshLineAtn: -----> {0}:
efmCuPmeThreshMinSnrMgn: -----> {0}:
efmCuPmeLineAtnCrossingEnable: ----> {false}:
efmCuPmeSnrMgnCrossingTrapEnable: -> {false}:
efmCuPmeDeviceFaultEnable: -----> {false}:
efmCuPmeConfigInitFailEnable: -----> {false}:
efmCuPmeProtocolInitFailEnable: ---> {false}:
efmCuPme2BProfileDescr: -----> {}:
efmCuPme2BRegion: -----> {region1}:
efmCuPme2BDataRate: -----> {0}:
efmCuPme2BPower: -----> {0}:
efmCuPme2BConstellation: -----> {adaptive}:
efmCuPme2BProfileRowStatus: -----> {active}:
efmCuPmeNtr: -----> {ntr-local-osc}:
efmCuPmeThreshMaxSnrMgnDelta: -----> {20}:
efmCuPmeMaintenanceMode: -----> {off}: automatic-once
efmCuPmeMaintenanceStartTime: -----> {00:00}: 3:00
efmCuPmeMaintenanceEndTime: -----> {23:59}: 4:00
efmCuPmeSnrMonitoringInterval: ----> {01:00}:
```



```
efmCuPmeErrorThreshMonEnable: -----> {false}
efmCuPmeErrorThreshMonNotifyEnable: -> {false}
efmCuPmeErrorThreshMonInterval: -----> {12}
efmCuPmeErrorThreshMonClrInterval: --> {1800}
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

In this case, the SHDSL line will retrain one time only between 3:00 am and 4:00 am if the SNR value is outside of the specified threshold.

3 To verify the changes, enter:

```
ZSH> get pme-profile 1-5-1-0/shdsl
pme-profile 1-5-1-0/shdsl
efmCuPmeAdminSubType: -----> {ieee2basetlr}
efmCuPmeAdminProfile: -----> {0}
efmCuPAFRemoteDiscoveryCode: -----> {}
efmCuPmeThreshLineAtn: -----> {0}
efmCuPmeThreshMinSnrMgn: -----> {0}
efmCuPmeLineAtnCrossingEnable: -----> {false}
efmCuPmeSnrMgnCrossingTrapEnable: ---> {false}
efmCuPmeDeviceFaultEnable: -----> {false}
efmCuPmeConfigInitFailEnable: -----> {false}
efmCuPmeProtocolInitFailEnable: -----> {false}
efmCuPme2BProfileDescr: -----> {}
efmCuPme2BRegion: -----> {region1}
efmCuPme2BDataRate: -----> {0}
efmCuPme2BPower: -----> {0}
efmCuPme2BConstellation: -----> {adaptive}
efmCuPme2BProfileRowStatus: -----> {active}
efmCuPmeNtr: -----> {ntr-local-osc}
efmCuPmeThreshMaxSnrMgnDelta: -----> {20}
efmCuPmeMaintenanceMode: -----> {automatic-once}
efmCuPmeMaintenanceStartTime: -----> {3:00}
efmCuPmeMaintenanceEndTime: -----> {4:00}
efmCuPmeSnrMonitoringInterval: -----> {01:00}
efmCuPmeErrorThreshMonEnable: -----> {false}
efmCuPmeErrorThreshMonNotifyEnable: -> {false}
efmCuPmeErrorThreshMonInterval: -----> {12}
efmCuPmeErrorThreshMonClrInterval: --> {1800}
```

4 For *automatic-daily* mode, set the **efmCuPmeMaintenanceMode** to *automatic-daily* and set the maintenance period start and end time in **efmCuPmeMaintenanceStartTime** and **efmCuPmeMaintenanceEndTime**.

```
ZSH> update pme-profile 1-5-2-0/shdsl
pme-profile 1-5-2-0/shdsl
Please provide the following: [q]uit.
efmCuPmeAdminSubType: -----> {ieee2basetlr}:
efmCuPmeAdminProfile: -----> {0}:
efmCuPAFRemoteDiscoveryCode: -----> {}:
efmCuPmeThreshLineAtn: -----> {0}:
```

```

efmCuPmeThreshMinSnrMgn: -----> {0}:
efmCuPmeLineAtnCrossingEnable: -----> {false}:
efmCuPmeSnrMgnCrossingTrapEnable: ---> {false}:
efmCuPmeDeviceFaultEnable: -----> {false}:
efmCuPmeConfigInitFailEnable: -----> {false}:
efmCuPmeProtocolInitFailEnable: -----> {false}:
efmCuPme2BProfileDescr: -----> {}:
efmCuPme2BRegion: -----> {region1}:
efmCuPme2BDataRate: -----> {0}:
efmCuPme2BPower: -----> {0}:
efmCuPme2BConstellation: -----> {adaptive}:
efmCuPme2BProfileRowStatus: -----> {active}:
efmCuPmeNtr: -----> {ntr-local-osc}:
efmCuPmeThreshMaxSnrMgnDelta: -----> {20}:
efmCuPmeMaintenanceMode: -----> {off}: automatic-daily
efmCuPmeMaintenanceStartTime: -----> {00:00}: 1:00
efmCuPmeMaintenanceEndTime: -----> {23:59}: 2:00
efmCuPmeSnrMonitoringInterval: -----> {01:00}:
efmCuPmeErrorThreshMonEnable: -----> {false}:
efmCuPmeErrorThreshMonNotifyEnable: -> {false}:
efmCuPmeErrorThreshMonInterval: -----> {12}:
efmCuPmeErrorThreshMonClrInterval: --> {1800}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

In this case the SHDSL line will automatically retrain every day between 1:00 am and 2:00 am if the SNR value is outside of the specified threshold.

- 5 For the *automatic-continuous* mode, set the **efmCuPmeMaintenanceMode** to *automatic-continuous*, set the **efmCuPmeMaintenanceStartTime** and the **efmCuPmeMaintenanceEndTime** for the time of the maintenance period, and set the monitoring interval in the **efmCuPmeSnrMonitoringInterval**. The monitoring interval determines how often during the maintenance period the system checks to retrain the SNR rate.

```

zSH> update pme-profile 1-5-3-0/shdsl
pme-profile 1-5-3-0/shdsl
Please provide the following: [q]uit.
efmCuPmeAdminSubType: -----> {ieee2basetlr}:
efmCuPmeAdminProfile: -----> {0}:
efmCuPAFRemoteDiscoveryCode: -----> {}:
efmCuPmeThreshLineAtn: -----> {0}:
efmCuPmeThreshMinSnrMgn: -----> {0}:
efmCuPmeLineAtnCrossingEnable: -----> {false}:
efmCuPmeSnrMgnCrossingTrapEnable: ---> {false}:
efmCuPmeDeviceFaultEnable: -----> {false}:
efmCuPmeConfigInitFailEnable: -----> {false}:
efmCuPmeProtocolInitFailEnable: -----> {false}:
efmCuPme2BProfileDescr: -----> {}:
efmCuPme2BRegion: -----> {region1}:

```

```

efmCuPme2BDataRate: -----> {0}:
efmCuPme2BPower: -----> {0}:
efmCuPme2BConstellation: -----> {adaptive}:
efmCuPme2BProfileRowStatus: -----> {active}:
efmCuPmeNtr: -----> {ntr-local-osc}:
efmCuPmeThreshMaxSnrMgnDelta: -----> {20}:
efmCuPmeMaintenanceMode: -----> {off}: automatic-continuous
efmCuPmeMaintenanceStartTime: -----> {00:00}: 1:00
efmCuPmeMaintenanceEndTime: -----> {23:59}: 7:00
efmCuPmeSnrMonitoringInterval: -----> {01:00}: 00:30
efmCuPmeErrorThreshMonEnable: -----> {false}:
efmCuPmeErrorThreshMonNotifyEnable: -> {false}:
efmCuPmeErrorThreshMonInterval: -----> {12}:
efmCuPmeErrorThreshMonClrInterval: --> {1800}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

In this example the maintenance mode is set to **automatic-continuous**, the start and end time are set from 1:00 am to 7:00 am, and the monitoring interval is set to every half hour.

Configure SNR crossing traps

An SNR crossing trap is sent only when the SNR value has gone from within range of the threshold to exceeding the range or from exceeding the range of the threshold to within the range. SNR monitoring will occur at the frequency configured by the interval parameter (**efmCuPmeSnrMonitoringInterval**).

Configuring SNR crossing traps

Set **efmCuPmeSnrMgnCrossingTrapEnable** to *true*.

Verify SNR monitoring is enabled/disabled

You can verify that SNR monitoring is disabled either by using the **snrmon show** command for the interface or checking the parameter **efmCuPmeMaintenanceMode** in the **pme-profile** for the interface.

When you use the **snrmon show** command and the *Mode* displays **off**, SNR monitoring is disabled.

```
zSH> snrmon show 1-5-1-0/shdsl
```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	0		20		00:00	23:59	01:00	disabled	off

If the *Mode* shows any other value than off, SNR monitoring is enabled.

```
zSH> snrmon show 1-5-1-0/shdsl
```

Shdsl Snr Monitoring									
Port	Snr	Min	Snr	Delta	Start	End	Interval	Trap	Mode
1	0		20		00:00	23:59	01:00	disabled	automatic-daily

G. SHDSL SNR monitoring example

A line which will retrain during the next maintenance interval, because the **DslLineSnrMgn** of 18 dB is greater than the sum of **efmCuTargetCurrentConditionSnrMgn** (6 dB) and **efmCuPmeThreshMaxSnrMgnDelta** (10 dB); all parameters involved in determining whether the retrain should occur are shown in **bold**.

In this example the maintenance period is for the entire day (from 00:00 to 23:59), so technically there is no next maintenance window since it is always within the maintenance window. The link will retrain every 10 minutes if **dslslnrmgn** continues to exceed the set threshold since the interval parameter (**efmCuPmeSnrMonitoringInterval**) is 10 minutes.

```
zSH> get pme-profile 1-7-6-0/shdsl
pme-profile 1-7-6-0/shdsl
efmCuPmeAdminSubType: -----> {ieee2basetlr}
efmCuPmeAdminProfile: -----> {0}
efmCuPAFRemoteDiscoveryCode: -----> {}
efmCuPmeThreshLineAtn: -----> {0}
efmCuPmeThreshMinSnrMgn: -----> {0}
efmCuPmeLineAtnCrossingEnable: -----> {true}
efmCuPmeSnrMgnCrossingTrapEnable: -> {true}
efmCuPmeDeviceFaultEnable: -----> {false}
efmCuPmeConfigInitFailEnable: -----> {false}
efmCuPmeProtocolInitFailEnable: ---> {false}
efmCuPme2BProfileDescr: -----> {}
efmCuPme2BRegion: -----> {region1}
efmCuPme2BDataRate: -----> {0}
efmCuPme2BPower: -----> {0}
efmCuPme2BConstellation: -----> {adaptive}
efmCuPme2BProfileRowStatus: -----> {active}
efmCuPmeNtr: -----> {ntr-local-osc}
efmCuPmeThreshMaxSnrMgnDelta: -----> {10}
efmCuPmeMaintenanceMode: -----> {automatic-continuous}
efmCuPmeMaintenanceStartTime: -----> {00:00}
efmCuPmeMaintenanceEndTime: -----> {23:59}
efmCuPmeSnrMonitoringInterval: -----> {00:10}
```

```
zSH> get efm-port 1-7-6-0/shdsl
efmCuPAFAdminState: -----> {enabled}
efmCuPAFDiscoveryCode: -----> {}
efmCuAdminProfile: -----> {0x01}
efmCuTargetDataRate: -----> {50000}
efmCuTargetWorstCaseSnrMgn: -----> {0}
efmCuThreshLowBandwidth: -----> {0}
efmCuLowBandwidthEnable: -----> {false}
efmCuTargetCurrentConditionMode: ---> {true}
efmCuTargetCurrentConditionSnrMgn: -> {6}
```

```
efmCuTargetWorstCaseMode: -----> {false}
```

Use the **dslstat** command to verify the DSL Line Margin (DslLineSnrMgn) in bold which is displayed in tenths of a dB, so it is 18 dB

```
zSH> dslstat 1-7-6-0/shdsl
```

General Stats:

```
-----
AdminStatus.....UP
LineStatus.....DATA
Line uptime
(DD:HH:MM:SS).....0:00:00:47
DslUpLineRate (bitsPerSec).....5696000
DslDownLineRate (bitsPerSec).....5696000
DslMaxAttainableUpLineRate (bitsPerSec).....5696000
DslMaxAttainableDownLineRate (bitsPerSec).....5696000
Out Octets.....489212764
Out Pkts/Cells.....1913588
Out Discards.....0
Out Errors.....0
In Octets.....12034672
In Pkts/Cells.....161629
In Discards.....0
In Errors.....0
```

DSL Physical Stats:

```
-----
DslLineSnrMgn (tenths dB).....180
DslLineAtn (tenths dB).....0
DslCurrOutputPwr (tenths dB).....0
LOFS.....0
LOLS.....0
LOSS.....24
ESS.....1618
CRC Errors.....42
Inits.....139
```

A line which will NOT retrain because the **DslLineSnrMgn** of 2 dB is greater than the **efmCuPmeThreshMinSnrMgn** of 0 dB, and the **DslLineSnrMgn** of 2 dB is less than the sum of **efmCuTargetWorstCaseSnrMgn** (0 dB) and the **efmCuPmeThreshMaxSnrMgnDelta** (3 dB); all parameters involved in determining whether the retrain should occur are shown in **bold**.

```
zSHc> get pme-profile 1-7-4-0/shdsl
```

```
pme-profile 1-7-4-0/shdsl
efmCuPmeAdminSubType: -----> {ieee2basetlr}
efmCuPmeAdminProfile: -----> {0}
efmCuPAFRemoteDiscoveryCode: -----> {}
efmCuPmeThreshLineAtn: -----> {0}
efmCuPmeThreshMinSnrMgn: -----> {0}
efmCuPmeLineAtnCrossingEnable: ----> {false}
efmCuPmeSnrMgnCrossingTrapEnable: -> {true}
```

```

efmCuPmeDeviceFaultEnable: -----> {false}
efmCuPmeConfigInitFailEnable: -----> {false}
efmCuPmeProtocolInitFailEnable: ---> {false}
efmCuPme2BProfileDescr: -----> {}
efmCuPme2BRegion: -----> {region1}
efmCuPme2BDataRate: -----> {0}
efmCuPme2BPower: -----> {0}
efmCuPme2BConstellation: -----> {adaptive}
efmCuPme2BProfileRowStatus: -----> {active}
efmCuPmeNtr: -----> {ntr-local-osc}
efmCuPmeThreshMaxSnrMgnDelta: -----> {3}
efmCuPmeMaintenanceMode: -----> {automatic-continuous}
efmCuPmeMaintenanceStartTime: -----> {00:00}
efmCuPmeMaintenanceEndTime: -----> {23:59}
efmCuPmeSnrMonitoringInterval: ----> {01:00}

```

```

zSH> get efm-port 1-7-4-0/shdsl
efm-port 1-7-4-0/shdsl
efmCuPAFAdminState: -----> {enabled}
efmCuPAFDiscoveryCode: -----> {}
efmCuAdminProfile: -----> {0x01}
efmCuTargetDataRate: -----> {50000}
efmCuTargetWorstCaseSnrMgn: -----> {0}
efmCuThreshLowBandwidth: -----> {0}
efmCuLowBandwidthEnable: -----> {false}
efmCuTargetCurrentConditionMode: ---> {false}
efmCuTargetCurrentConditionSnrMgn: -> {6}
efmCuTargetWorstCaseMode: -----> {true}

```

[Table82](#) describes the error monitoring parameters in the **pme-profile**.

Table 82: Error monitoring parameters in the pme-profile

efmCuPmeErrorThreshMonEnable	Enables/disables the error threshold monitoring. Error threshold monitoring looks for errors on the physical line. When there have been efmCuPmeErrorThreshMonInterval number of consecutive seconds with errors, the line will first be taken down with a retrain in hopes of bettering the SNR to rectify the error situation. If, when the line comes up, we do not achieve efmCuPmeErrorThreshMonClrInterval number of consecutive error-free seconds before hitting another efmCuPmeErrorThreshMonInterval number of consecutive seconds with errors, the line will cease to be used for carrying traffic. The physical line will remain active so error monitoring can continue but data will no longer traverse until efmCuPmeErrorThreshMonClrInterval number of consecutive error-free seconds is achieved, at which time the line will resume carrying traffic.
efmCuPmeErrorThreshMonNotifyEnable	Enables/disables the error threshold monitoring notification (via CLI or alarm manager) when an error threshold has been exceeded or cleared.

Table 82: Error monitoring parameters in the pme-profile (Continued)

efmCuPmeErrorThreshMonInterval	Specifies the number of consecutive seconds of detecting errors that, once reached, will cause the physical line to be deemed a poor performer and causes an action to be taken. See efmCuPmeErrorThreshMonEnable .
efmCuPmeErrorThreshMonClrInterval	Specifies the number of consecutive error-free seconds that must be achieved in order to declare the physical line usable for carrying data traffic. See efmCuPmeErrorThreshMonEnable .

Compare the with the **DslLineSnrMgn** line statistics using the **dslstat** command:

```
zSH> dslstat 1-7-4-0/shdsl
```

General Stats:

```
-----
AdminStatus.....UP
LineStatus.....DATA
Line uptime(DD:HH:MM:SS).....0:19:13:44
DslUpLineRate (bitsPerSec).....2576000
DslDownLineRate (bitsPerSec).....2576000
DslMaxAttainableUpLineRate (bitsPerSec).....5696000
DslMaxAttainableDownLineRate (bitsPerSec)....5696000
Out Octets.....9372
Out Pkts/Cells.....64
Out Discards.....0
Out Errors.....0
In Octets.....7961682
In Pkts/Cells.....241214
In Discards.....0
In Errors.....411809
```

DSL Physical Stats:

```
-----
DslLineSnrMgn (tenths dB).....20
DslLineAtn (tenths dB).....210
DslCurrOutputPwr (tenths dB).....0
LOFS.....0
LOLS.....0
LOSS.....0
ESS.....229
CRC Errors.....0
Inits.....11
```

Table 83: SHDSL statistics

General statistics	Description
AdminStatus	Administrative status of the port: Values: Up Interface is ready to pass packets. Down Interface is unable to pass packets. Testing Interface is in a special testing state and is unable to pass packets.
LineStatus	Line status provides information about the SHDSL link. Values for a single SHDSL line: ACT — the line currently has link and can pass traffic in both directions OOS — the line does not have link TRAFFIC DISABLE — The line currently has link but not underlying SHDSL protocol; traffic will not pass.
Line uptime (DD:HH:MM:SS)	How long the interface has been up in dd hh mm (day, hour, minute, second) format.
DslUpLineRate (bitsPerSec)	Displays the DSL upstream (customer premise > central office) line rate on this interface.
DslDownLineRate (bitsPerSec)	Displays the DSL downstream (central office > customer premise) line rate on this interface.
DslMaxAttainableUpLineRate (bitsPerSec)	Displays the maximum line rate that can be supported on this line in the upstream direction.
DslMaxAttainableDownLineRate (bitsPerSec)	Displays the maximum line rate that can be supported on this line in the downstream direction.
Out Octets	Number of transmitted octets.
Out Pkts/Cells	Number of transmitted packets/cells
Out Discards	Number of transmission discards.
Out Errors	Number of transmission errors.
In Octets	Number of received octets.
In Pkts/Cells	Number of transmitted packets/cells
In Discards	Number of received discards.
In Errors	Number of receive errors.
SHDSL Physical Stats:	
DslLineSnrMgn (tenths dB)	DSL Line Signal to Noise Ratio Margin — The strength of the DSL signal relative to the noise on line.

Table 83: SHDSL statistics (Continued)

General statistics	Description
DslLineAtn (tenths dB)	DSL Line Attenuation — Measure of the signal degradation between the SHDSL port and the modem.
DslCurrOutputPwr (tenths dB)	Not currently used.
LOFS	Number of Loss of Frame Seconds.
LOLS	Number of Loss of Line Seconds.
LOSS	Number of Loss of Signal Seconds.
ESS	Number of errored seconds (the number of one-second intervals containing one or more CRC anomalies or one or more LoS or Sef defects) that has been reported in the current 15-minute interval.
CRC Errors	Cyclic Redundancy Check Errors — CRC Checks for transmission errors. The CRC code is computed from the data in the message. If the data is altered the CRC computation will not be in agreement with the data.
Initis	Number of line initialization attempts, including both successful and failed attempts.

Disable SNR monitoring

To disable SNR monitoring for a link use the **snrmon modify** command with **mode** set to **off**.

```
zSH> snrmon modify 1-5-1-0 mode off
```

SHDSL error monitoring

This section describes:

- [SHDSL error monitoring statistics, page689](#)
- [SHDSL error monitoring fields, page690](#)

SHDSL error monitoring statistics

The MXK provides the **errmon** command to view SHDSL error monitoring information.

Enter the **errmon stats interface/type** command to view SHDSL error monitoring statistics:

```
zSH> errmon stats 1-7-3-0/shdsl
```

```

Shdsl Error Monitoring Stats
Max
Port  TC  Down  CRC   ES   SES   Err Sec  Restart  Line Status
3      0    16  32427  32416      0      0      ACT
```

[Table 84](#) defines the **errmon stats** fields.

Table 84: Definitions of displayed statistics

Parameter	Description
TC Down	Count of how many times the TC layer went down since the physical link was obtained.
CRC	Count of CRC anomalies.
ES	Count of one second intervals during which one or more CRCs are reported.
SES	Count of one second intervals during which at least 50 CRCs are reported.
Max Errored Sec	Maximum consecutive seconds with errors without causing action to be taken by errmon features.
Restart	Count of the number of times the port was restarted by errmon features.
Line Status	Current status of the port.

SHDSL error monitoring fields

The **errmon interface/type** command monitors the physical interface, for example, *1-5-1-0/shdsl*.

Enter **errmon show** to view the SHDSL error monitoring fields and enter **errmon modify** to enable the notify and monitor fields and to modify the default values.

```
zSH> errmon show 1-7-3-0/shdsl
      Shdsl Error Monitoring
Port  Monitor  Notify  ErrorInt  ClearInt  Line Status
  3    disabled disabled      12      1800      ACT
```

The *Monitor* and the *Notify* fields must be enabled from the CLI:

Enable *Monitor*:

```
zSH> errmon modify 1-7-3-0/shdsl monitor enable
```

```
zSH> errmon show 1-7-3-0/shdsl
      Shdsl Error Monitoring
Port  Monitor  Notify  ErrorInt  ClearInt  Line Status
  3    enabled disabled      12      1800      ACT
```

Enable *Notify*:

```
zSH> errmon modify 1-7-3-0/shdsl notify enable
```

```
zSH> errmon show 1-7-3-0/shdsl
      Shdsl Error Monitoring
Port  Monitor  Notify  ErrorInt  ClearInt  Line Status
  3    enabled  enabled      12      1800      ACT
```

To change *errinterval* values enter:

```
zSH> errmon modify 1-7-3-0/shdsl errinterval 10
```

```
zSH> errmon show 1-7-3-0/shdsl
```

```

      Shdsl Error Monitoring
Port  Monitor  Notify  ErrorInt  ClearInt  Line Status
  3    enabled  enabled    10      1800      ACT

```

To change *clrinterval* values enter:

```
zSH> errmon modify 1-7-3-0/shdsl clrinterval 600
```

```
zSH> errmon show 1-7-3-0/shdsl
```

```

      Shdsl Error Monitoring
Port  Monitor  Notify  ErrorInt  ClearInt  Line Status
  3    enabled  enabled    10       600      ACT

```

Table 85: errmon parameters

Parameter	Description
link interface (port)	Name and type of a physical interface. For example, 1-5-1-0/shdsl.
monitor	Determines if error monitoring should be performed on the link.
notify	Determines if a notification to the CLI, alarm manager, and ZMS should be generated when an error threshold is exceeded or cleared on the link.
errinterval	Specifies the number of consecutive seconds of detecting errors that, once reached, causes the physical line to be considered a poor performer and action to be taken.
clrinterval	Specifies the number of consecutive error-free seconds that must be achieved in order to declare the physical line usable for transporting data traffic.

errmon modify

Syntax `error modify|show|stats link/interface`

SHDSL statistics

Verifying the interface

Use the **dslstat** command to display the status of an SHDSL interface:

```
zSH> dslstat 1-7-4-0/shdsl
```

General Stats:

```

-----
AdminStatus.....UP
LineStatus.....DATA
Line uptime (DD:HH:MM:SS).....0:04:14:45
DslUpLineRate (bitsPerSec).....832000
DslDownLineRate (bitsPerSec).....832000
DslMaxAttainableUpLineRate (bitsPerSec).....5696000
DslMaxAttainableDownLineRate (bitsPerSec)....5696000
Out Octets.....311851068
Out Pkts/Cells.....1308813
Out Discards.....0
Out Errors.....0
In Octets.....307992514
In Pkts/Cells.....1294816
In Discards.....0
In Errors.....0

```

DSL Physical Stats:

```

-----
DslLineSnrMgn (tenths dB).....80
DslLineAtn (tenths dB).....280
DslCurrOutputPwr (tenths dB).....0
LOFS.....0
LOLS.....9
LOSS.....67
ESS.....177
CRC Errors.....16
Inits.....15

```

[Table86](#) defines the statistics displayed in the **dslstat** command for an SHDSL line.

Clearing DSL counters (SHDSL)

You can clear DSL counters to make identifying the changing statistics easier to read.

- 1 Clear the statistics using the **dslstat clear** command

```
zSH> dslstat clear 1-7-4-0/shdsl
```

- 2 View the changes

For reference the **dslstat** command ([Use the dslstat command to display the status of an SHDSL interface: on page691](#)) shows the statistics prior to clearing the statistics. Statistic which are cleared by the **dslstat clear** command are in **bold**.

```
zSH> dslstat clear 1-7-4-0/shdsl
zSH> dslstat 1-7-4-0/shdsl
```

General Stats:

```

-----
AdminStatus.....UP

```

```

LineStatus.....DATA
Line uptime(DD:HH:MM:SS).....0:04:14:58
DslUpLineRate (bitsPerSec).....832000
DslDownLineRate (bitsPerSec).....832000
DslMaxAttainableUpLineRate (bitsPerSec).....5696000
DslMaxAttainableDownLineRate (bitsPerSec)....5696000
Out Octets.....0
Out Pkts/Cells.....0
Out Discards.....0
Out Errors.....0
In Octets.....0
In Pkts/Cells.....0
In Discards.....0
In Errors.....0

DSL Physical Stats:
-----
DslLineSnrMgn (tenths dB).....80
DslLineAtn (tenths dB).....280
DslCurrOutputPwr (tenths dB).....0
LOFS.....0
LOLS.....0
LOSS.....0
ESS.....0
CRC Errors.....0
Inits.....0

```

Table 86: SHDSL statistics

General statistics	Description
AdminStatus	Administrative status of the port: Values: Up Interface is ready to pass packets. Down Interface is unable to pass packets. Testing Interface is in a special testing state and is unable to pass packets.
LineStatus	Line status provides information about the SHDSL link. For information about the status on EFM or N2N bond groups, see Bond group statistics and port statistics on page 695 . Values for a single SHDSL line: ACT — the line currently has link and can pass traffic in both directions OOS — the line does not have link TRAFFIC DISABLE — The line currently has link but not underlying SHDSL protocol; traffic will not pass.
Line uptime (DD:HH:MM:SS)	How long the interface has been up in dd hh mm (day, hour, minute, second) format.

Table 86: SHDSL statistics (Continued)

General statistics	Description
DslUpLineRate (bitsPerSec)	Displays the DSL upstream (customer premise > central office) line rate on this interface.
DslDownLineRate (bitsPerSec)	Displays the DSL downstream (central office > customer premise) line rate on this interface.
DslMaxAttainableUpLineRate (bitsPerSec)	Displays the maximum line rate that can be supported on this line in the upstream direction.
DslMaxAttainableDownLineRate (bitsPerSec)	Displays the maximum line rate that can be supported on this line in the downstream direction.
Out Octets	Number of transmitted octets.
Out Pkts/Cells	Number of transmitted packets/cells
Out Discards	Number of transmission discards.
Out Errors	Number of transmission errors.
In Octets	Number of received octets.
In Pkts/Cells	Number of transmitted packets/cells
In Discards	Number of received discards.
In Errors	Number of receive errors.
SHDSL Physical Stats:	
DslLineSnrMgn (tenths dB)	DSL Line Signal to Noise Ratio Margin — The strength of the DSL signal relative to the noise on line.
DslLineAtn (tenths dB)	DSL Line Attenuation — Measure of the signal degradation between the SHDSL port and the modem.
DslCurrOutputPwr (tenths dB)	Not currently used.
LOFS	Number of Loss of Frame Seconds.
LOLS	Number of Loss of Line Seconds.
LOSS	Number of Loss of Signal Seconds.
ESS	Number of errored seconds (the number of one-second intervals containing one or more CRC anomalies or one or more LoS or Sef defects) that has been reported in the current 15-minute interval.
CRC Errors	Cyclic Redundancy Check Errors — CRC Checks for transmission errors. The CRC code is computed from the data in the message. If the data is altered the CRC computation will not be in agreement with the data.
Inits	Number of line initialization attempts, including both successful and failed attempts.

Bond group statistics and port statistics

The MXK provides the ability to view statistics for both each port associated with a link in a bond group and an individual bond group.

- [View port statistics, page 695](#)
- [View bond group statistics, page 696](#)

View port statistics

EFM bonding fragments packets across multiple lines so that packet counts for each EFM port indicates the number of EFM packet fragments for that port. At the physical port level, unicast packet counts show the number of packet fragments for that port. Octets for the physical port include all bytes received, including those from errored packet fragments and protocol overhead.

Data in the **dslstat** command is provided for each port associated with a link in the bond group. The data is collected differently for N2N and EFM ports and bond groups.

View each of the 24 SHDLS ports by entering **dslstat shelf-slot-port-subport/type**:

```
zSH> dslstat 1-5-1-0/shdsl
General Stats:
-----
AdminStatus.....UP
LineStatus.....DATA
Line uptime (DD:HH:MM:SS).....0:04:56:27
DslUpLineRate (bitsPerSec).....5696000
DslDownLineRate (bitsPerSec).....5696000
DslMaxAttainableUpLineRate (bitsPerSec).....5696000
DslMaxAttainableDownLineRate (bitsPerSec)....5696000
Out Octets.....0
Out Pkts/Cells.....0
Out Discards.....0
Out Errors.....0
In Octets.....163790
In Pkts/Cells.....904
In Discards.....0
In Errors.....0
DSL Physical Stats:
-----
DslLineSnrMgn (tenths dB).....190
DslLineAtn (tenths dB).....10
DslCurrOutputPwr (tenths dB).....0
LOFS.....0
LOLS.....0
LOSS.....3
ESS.....149
CRC Errors.....0
Inits.....3
```

View bond group statistics

The MXK and other bonding capable devices provide the **bond stats** *shelf-slot-port-subport/type* command to display both the status of the bond group and the status of each individual link in the bond group and to provide statistics for the bond group. A bond group is the aggregate of individual links on a device connected to the same CPE that provides a higher bandwidth than individual links can provide.

To view the statistics for an MXK bond group enter **bond stats** *shelf-slot-port-subport/type*:

```
zSH> bond stats 1-6-201-0/efmbond
***** Bond group statistics *****
      Group Info
Slot      GrpId      Interface Name
  6         185      1-6-201-0/efmbond
AdminStatus      OperStatus      Bandwidth      Last Change
  UP              UP          45568000      0.04:14:53
Threshold Alarm Config
  disabled

      Group Members

Port      Admin      Oper      Bandwidth (tenths      Snr      Tc Layer      Interface Name
6/2        UP        UP        5696000      180        0      1-6-2-0/shdsl
6/1        UP        UP        5696000      180        0      1-6-1-0/shdsl
6/8        UP        UP        5696000      180        0      1-6-8-0/shdsl
6/7        UP        UP        5696000      170        0      1-6-7-0/shdsl
6/6        UP        UP        5696000      170        0      1-6-6-0/shdsl
6/5        UP        UP        5696000      180        0      1-6-5-0/shdsl
6/4        UP        UP        5696000      180        0      1-6-4-0/shdsl
6/3        UP        UP        5696000      180        0      1-6-3-0/shdsl

      Statistics (Received)
Octets      108011
Ucast        5
Mcast       737
Bcast       10
Discards      0
Errors        0

      Statistics (Transmitted)
Octets      2390
Ucast       10
Mcast        0
Bcast        5
Discards      0
```

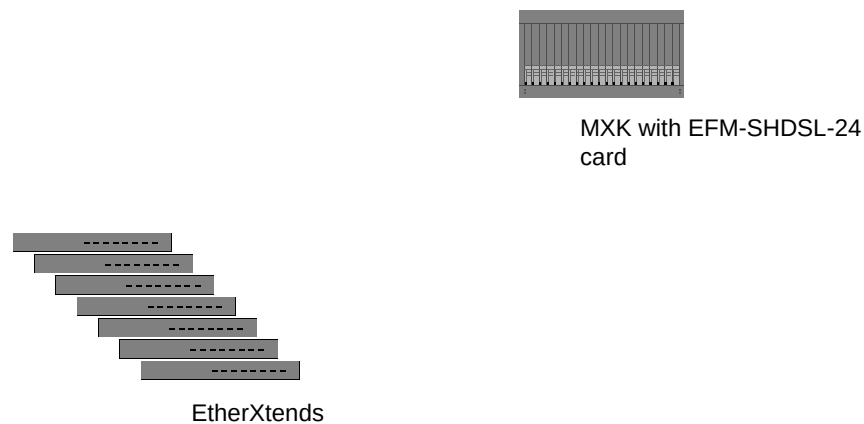
802.3ah EFM OAM

EFM OAM uses an in-band link layer OAM packet exchange between MXK EFM interfaces and OAM-capable CPEs, such as EtherXtend and EtherXtend Lite. The OAM-capable CPE functions as a remote peer to provide event

notification. The EFM and N2N bond groups are Ethernet-like interfaces and support EFM OAM.

When EFM OAM is configured on a MXK EFM or Ethernet-like interface in active mode, the discovery process is started. If the interface peer also has OAM enabled, the discovery process continues until the peer is located. If the discovery process does not find a peer, the active interface continues sending the initial Information OAMPDU once a second until a peer OAM-enabled CPE device responds.

EFM OAM supports the MXK-EFM-SHDSL-24, MXK-EFM-SHDSL-24NTWC, MXK-EFM-SHDSL-24NTP card interfaces connected to EtherXtend and other compatible CPEs.



Configuring OAM support

The OAM interface is defined by an **ether-oam** profile that specifies the options for active/passive mode, loopback, and notification for events. By default, OAM is disabled on all MXK uplink and EFM interfaces.

To configure OAM features:

- 1 Create a new OAM profile for the desired EFM interface. By default, this profile is in passive mode with loopback disabled.

This example configures EFM OAM in active mode on EFM bond group 1-4-50-0/efmbond on a EFM-SHDSL-24 card in slot 4.

```
zSH> eth-oam add 1-4-50-0/efmbond active
```

- 2 Create a new OAM profile for the desired EtherXtend interface. By default, this profile is in passive mode with loopback disabled.

This example configures EFM OAM in passive mode on EFM bond group 1-1-40-0/efmbond on the peer EtherXtend.

```
zSH> eth-oam add 1-1-40-0/efmbond passive
```

- 3 Enter commands to modify and display OAM parameters.

The **eth-oam modify** command provides access to configurable settings in the ether-oam profile.

The **eth-oam show** command displays configured OAM settings.

The **eth-oam stats** command displays OAM statistics for a specified physical interface or bond group or all OAM interfaces.

eth-oam add

Configures and enables OAM interface on a physical interface.

Syntax **eth-oam interface/type [active | passive]**

Options **interface/type**

Name and type of the physical interface or bond group.

active

Sets OAM to active mode on this interface. The default is passive.

passive

Sets OAM to passive mode on this interface. The default is passive.

eth-oam delete

Deletes and disables the OAM configuration on the specified physical interface. This command does not delete any other configurations on this interface such as bond groups and bridge interfaces.

Syntax **eth-oam delete interface/type**

Options **interface/type**

Name and type of the physical interface or bond group.

eth-oam modify

Modifies a configured eth-oam interface.

Syntax **eth-oam modify interface/type [active | passive]**

Options **interface/type**

Name and type of the physical interface or bond group.

eth-oam show

Displays configured OAM parameters for the specified interface. If no interface is specified, configured OAM parameters are displayed for all OAM enabled interfaces.

Syntax **eth-oam show interface/type [peer]**

Options **interface/type**

Name and type of the physical interface or bond group.

peer

Displays the learned configuration information of the peer for the given interface. Includes peer MAC address, peer vendor OUI, peer vendor unique info, peer mode, peer max OAM PDU size, peer configuration revision, peer supported functions.

eth-oam stats

Displays OAM statistics for the specified interface. If no option is specified, statistics are displayed for all OAM interfaces.

Syntax `eth-oam stats interface/type`

Options `interface/type`

Name and type of the physical interface or bond group.

MXK-EFM-SHDSL-24 pinouts

The MXK-EFM-SHDSL-24 cards use standard RJ-21X pinouts. [Table87](#) lists the port pinouts.

Table 87: MXK-EFM-SHDSL-24 Pinouts

Port	Pin Ring	Pin Tip
1126		
2227		
3328		
4429		
5530		
6631		
7732		
8833		
9934		
10	10	35
11	11	36
12	12	37
13	13	38
14	14	39
15	15	40

Table 87: MXK-EFM-SHDSL-24 Pinouts (Continued)

Port	Pin Ring	Pin Tip
16	16	41
17	17	42
18	18	43
19	19	44
20	20	45
21	21	46
22	22	47
23	23	48
24	24	49



Note: Pins 25 and 50 are not used.

Power and data connections for SHDSL CPE devices

This section describes the power connections on the MXK-EFM-SHDSL-24-NTP card that enable power to be delivered to Zhone SHDSL CPE devices and includes:

- [Deliver power and data to the CPE, page 700](#)
- [Enable power on the SHDSL line, page 702](#)

Deliver power and data to the CPE

The MXK-EFM-SHDSL-24-NTP card delivers power and data on the same wires. To deliver power to the CPE, two pairs of wires (four wires total) are required.

The specifications for the cables delivering power are as follows:

- 2 wires per port
- 26AWG (0.4 mm) or 24AWG (0.5 mm)

The LP IN port on the MXK-EFM-SHDSL-24-NTP card provides 12 pairs of wires to deliver power. The power is combined with the data and sent out over the 24 SHDSL ports to downstream CPE devices. One MXK-EFM-SHDSL-24-NTP card can provide power and data for up to 12 SHDSL CPE devices.

Figure57 illustrates the wiring connections for power and data being transmitted over the same pair of wires to a single CPE port. To power multiple CPE devices, use the pinouts described in Table87 to match SHDSL ports to the power pairs. Each set of four pins can power a single SHDSL CPE.

Figure 57: Example power and data delivered over the same wire pairs

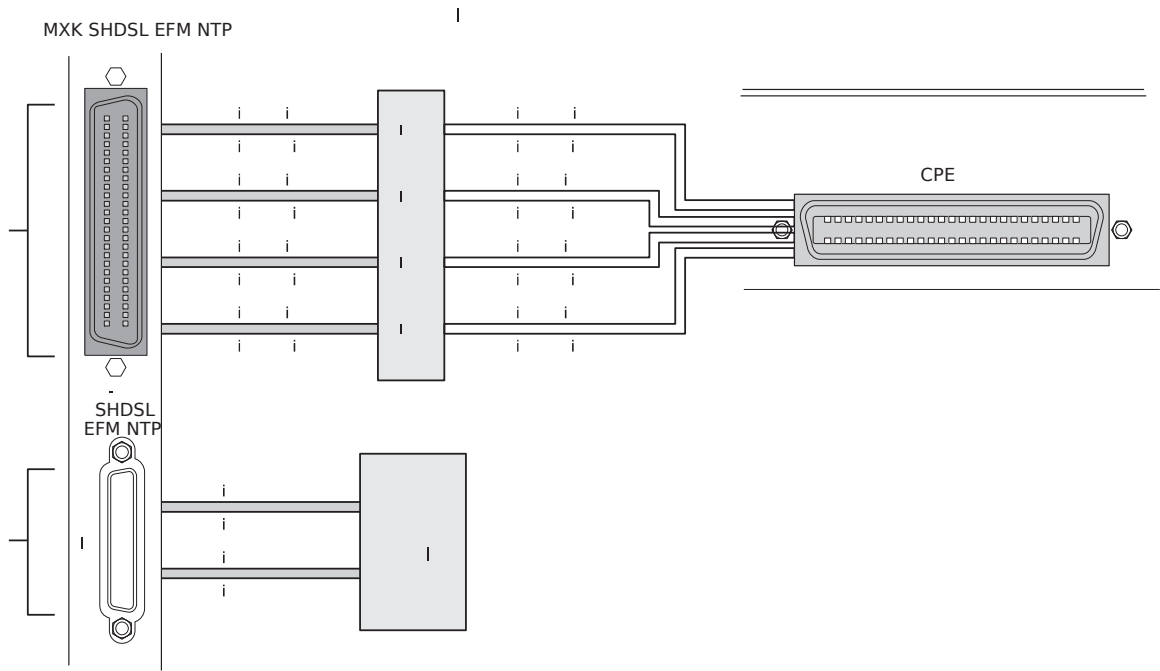


Table88 describes the power connections for the 26 pin connector.

Table 88: Power connections between the line power pins and the SHDSL ports

Line power pin	SHDSL
pin 1	port 1
pin 14	port 2
pin 2	port 3
pin 15	port 4
pin 3	port 5
pin 16	port 6
pin 4	port 7
pin 17	port 8
pin 5	port 9
pin 18	port 10

Table 88: Power connections between the line power pins and the SHDSL ports

Line power pin	SHDSL
pin 6	port 11
pin 19	port 12
pin 7	port 13
pin 20	port 14
pin 8	port 15
pin 21	port 16
pin 9	port 17
pin 22	port 18
pin 10	port 19
pin 23	port 20
pin 11	port 21
pin 24	port 22
pin 12	port 23
pin 25	port 24

Enable power on the SHDSL line

The line power feature is enabled by default. This means that voltage is applied to the SHDSL line by default. This voltage comes from an external power supply as shown in [Figure57](#). If the external power supply is not connected or turned off, voltage will simply not be supplied to the SHDSL line. However, the data stream will continue to be sent.

If someone needs to work on the line, voltage is removed from that line by setting adminstatus to maintenance. Maintenance mode stops the data stream and the voltage.



Note: When a port is set to maintenance mode, any MTAC testing that may be running on any port is turned off. Maintenance mode always has top priority.

```

zSH> update if-translate 1-13-1-0/shdsl
if-translate 1-13-1-0/shdsl
ifIndex: -----> {1421}
shelf: -----> {1}
slot: -----> {13}
port: -----> {1}
subport: -----> {0}
type: -----> {shdsl}
adminstatus: -----> {up} maintenance

```

```
physical-flag: -----> {true}  
iftype-extension: --> {none}  
ifName: -----> {1-13-1-0}  
redundancy-param1: -> {0}
```



Note: The SHDSL line power feature requires that two lines are used together and both must be set to up in the adminstatus field. The lines do not need to be adjacent.

MTAC testing

The line power feature on the MXK-EFM-SHDSL-24-NTP card is mutually exclusive with MTAC testing and takes precedence over MTAC. When the line power feature is being used, MTAC testing cannot occur. To run MTAC testing, no ports on the MXK-EFM-SHDSL-24-NTP card can be in maintenance mode.

15 MXK METALLIC TEST ACCESS (MTAC) CARDS

This chapter describes the MXK Metallic Test Access (MTAC) cards and explains how to configure it. The chapter includes:

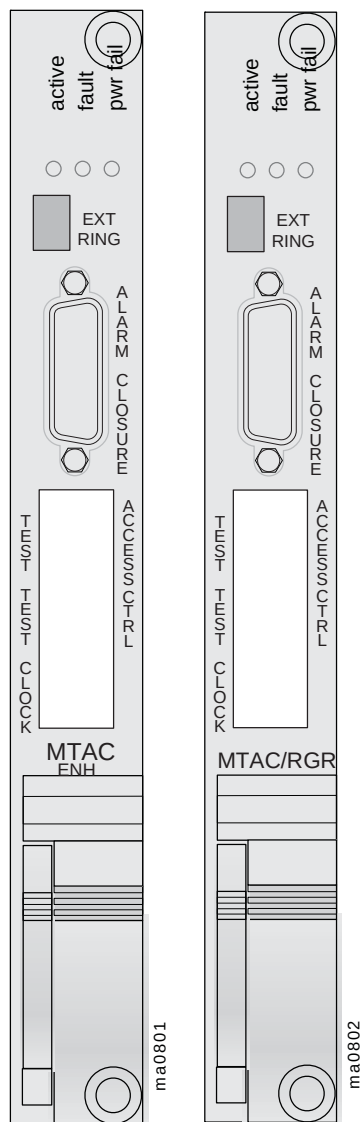
- [MTAC cards, page706](#)
- [Configure MTAC cards, page711](#)
- [Perform line testing using MTAC cards with external testing set, page713](#)
- [Perform internal line testing with MTAC/RING-ENH card, page718](#)
- [Configure external alarms, page742](#)
- [Configure an external clock, page743](#)
- [Connect an external ring source, page744](#)
- [MTAC cards pinouts, page746](#)

MTAC cards

This section describes the MXK MTAC cards and how to use them including:

- [MTAC card overview, page706](#)
- [MTAC card specifications, page708](#)
- [Connectors on the MTAC cards, page709](#)
- [Metallic loop testing, page709](#)
- [Internal look out line test, page710](#)
- [Ring generator, page710](#)

MTAC card overview



The MTAC cards supported in MXK are:

- The MXK-MTAC/RING card is a single slot card that supports metallic loop testing for DSL and POTS interfaces with the external test set.

Note that the type of tests provided will vary, depending on the type of card being tested.

It also supports external alarm inputs (12 circuits, wet or dry, normally open or normally closed), T1/E1 or BITS external network clock access, and ring generation (internal ring generator or access for an external ring generator).

- The MXK-MTAC/RING-ENH card is a single slot card that is the enhance version of the MXK-MTAC/RING card. In addition to the features in the MXK-MTAC/RING card, it also supports look-out internal line test (i.e. metallic loop testing without external test set).



Note: The MTAC cards supported in MXK are same MTAC cards supported in MALC. The only difference is they have the different product name:

In MXK, they are MXK-MTAC/RING and MXK-MTAC/RING-ENH.

In MALC, they are MALC-MTAC/RING and MALC-MTAC-RING-ENH.



Note: The MXK supports only one active MTAC card at a time and a total of two MTAC cards in the system.

The MTAC cards provide metallic test access to verify the local loop conditions, perform line testing on distant regions of the physical copper cable connecting the MXK and remote devices. It can assess breakages in the cable, identifying the following data:

- **Distance.** Identifies the amount of distance between the MTAC card and the location of the break or open that occurred on the copper cable.
- **Shorts.** Identifies the port to which a cable containing an electrical short is connected.
- **Unbalance.** Identifies if one side is longer between the tip and the ring, creating an unbalance in the connection.
- **Metallic noise.** Identifies any impairments on the cable that indicate an interruption on the ring.

MTAC card specifications

Table 89: MTAC card specifications

Specification	Value
Size	1 slot
Physical interfaces	• Metallic test access port: An RJ45 connector that connects to the external test set. It connects the external test set to metallic test bus on backplane (supports one port test simultaneously in system). • External test set control port: A serial control RS232D signalling port on RJ45 connector that provides a control connection to the external test set. • External clock input port: An RJ45 connector that accepts T1/E1 or BITS external clock reference (all versions), provisionable as system clock source. • External ring generator input port: A two position plug spaced at 5.08mm conforming to the IEC 60664-1 industry standard, such as the RIA Type 249 part number 312491 02. This port connects to the external ring generator. • External alarm connector: A 26 pin D sub connector that supports 12 alarm closures for detecting various alarm types from collocated equipment. Supports isolated closure, ground and -48VDC closure (states and names provisionable in software).
Metallic test functions	Look-out testing (toward the loop) for ADSL, ULC, and POTS interfaces (with the exception of ADSL 32 cards).  Note: The type of tests provided will vary, depending on the type of card being tested.
Ring generation	External ring generator voltage connector. Internal ring voltage sine wave generator: 15 REN @ 86VRMS for MTAC/RING card. 25 REN @ 93VRMS for MALC-MTAC/RING-ENH card.
Redundancy	1+1 card redundancy
Clocking	All MTAC cards can be configured to use T1, E1, or 2 MHz signal as the clock source. The clocking reference on the MTAC card with 2 MHz BITS clock complies with ITU-T G.703 standard.
Accuracy field	-10% to +10%
Power consumption	8 W nominal, 38 W maximum at full ringing load for MTAC/RING card. 13 W nominal, 53 W maximum at full ringing load for MTAC/RING-ENH.

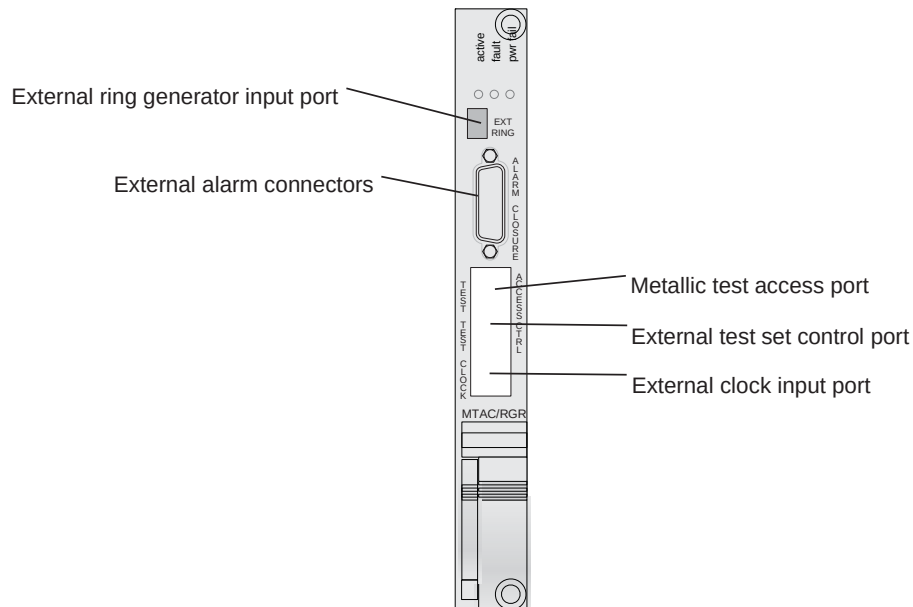
Connectors on the MTAC cards

All MTAC cards have following connectors:

- Metallic test access port
- External test set control port
- External clock input port
- External ring generator input port
- External alarm connectors

[Figure58](#) shows the connectors on the MTAC/RING card.

Figure 58: Connectors on MTAC/RING card



Metallic loop testing

The MTAC cards support metallic loop testing for T1, POTS, and DSL loops, providing preventive measures for potential line breaks.

All MTAC cards support external test sets. External test sets supported include Tollgrade, Harris/Fluke, and Teradyne 4-Tel components.

The MTAC/RING-ENH card also provides internal look-out line testing.

Internal look out line test

Internal line testing is supported by the MTAC/RING-ENH card. With its own integrated test set, the MTAC/RING-ENH card in each shelf can perform test out session without the external test set.

Cards supporting look-out test access

The MTAC cards provide access to external test equipment through an RJ45 connector for look-out test access. All ADSL-48 and EFM cards support look-out test access. The following table provides examples of common instances of these card types.

Table 90: Examples of common cards supporting look-out test access

Card Type	Example
ADSL-48	MXK-ADSL2+BCM-48A
	MXK-ADSL2+SPLT600-BCM-48A-2S
	MXK-ADSL2+SPLT900-BCM-48A-2S
	MXK-ADSL2+-POTS-BCM-48A-2S
EFM	MXK-EFM-SHDSL-24-NTWC
	MXK-EFM-SHDSL-24-NTP

Ring generator

The MTAC cards contain the ring generator for ADSL POTS combo card (i.e. MXK-ADSL2+-POTS-BCM-48A-2S) installed in the MXK. Ringing voltage is supplied to installed ADSL POTS combo card via a backplane bus. Note that only one MTAC card can supply ringing voltage to the system at a time.

The MTAC cards also contain a ringing voltage detector that senses the absence or false of ringing voltage on the card itself, or on an external ringing generator (if one exists).

- If the ringing voltage detector detects the absence of ringing voltage, a “Ringer source not detected” error message will be generated. The redundant MTAC card can supply the ringing voltage, or the MXK can be configured to use another external ringing generator.



Note: The MXK ground wires must be tied to the +48V battery return at the main power Distribution Center. Absence of this connection can cause malfunctions on some cards, including generation of the MTAC/RING-ENH card error message “Ringer source not detected”.

- If the ringing voltage exceeds the limit, the ringer voltage will be turned off, and a “Internal ringer disabled” error message will be generated. Software will attempt to restart it every 1 second. When the ring load drops back to normal, the MTAC internal ringer will automatically recover after 2 seconds, and a message “Ringer source detected” will be generated.

Configure MTAC cards

Caution: Each MTAC card in a redundant pair must be configured identically; the cards do not share state or configuration information. In addition, the user must manually keep the configuration of the active and standby cards in sync. This applies to both a matched pair and a mixed pair of MTAC cards.

Each MTAC card installed in the system must have a **card-profile**. Each type of slot card requires different settings in the **card-profile**.

MTAC cards have the following types and software images:

Table 91: MTAC card types

Card	Type	Name of software image
MXK-MTAC/RING	5003	<i>mtacring.bin</i>
MXK-MTAC/RING-ENH	5072	<i>tacitmring.bin</i>

Adding MTAC cards

The **card-profile** for MTAC cards require that the **card-line-type** (which specifies the external clock source type, *ds1* or *e1*) and **group card-group-id** (which is mandatory for the card has redundancy capability) are specified. Note that if there is another MTAC card is already added in the system, it ignores whatever **group card-group-id** you specify for the new MTAC card, and automatically assigns the card-group-id based on what card-group-id the other MTAC card is using.

To configure a redundant MTAC/RING or MTAC/RING-ENH card, create a second **card-profile** for the redundant card.

To add an MTAC card to the system:

- 1 Load the MTAC card image file on the MXK.
- 2 Install the MTAC card in the desired line card slot.
- 3 Create a **card-profile** for the card:

```
zSH> card add 15 linetype ds1 group 2
```

```
An autogenerated card-group-id [2] is assigned for this card type.
new card-profile 1/15/5003 added, sw-file-name "mtacring.bin", 2 options:
card-group-id 2 card-line-type ds1
```

After performing a **card add** in a slot, the slot resets and begins downloading the software image from the flash card. This could take a few moments.

When the card has finished loading, a log message similar to the following is displayed (if logging is enabled):

```
zSH> Card in slot slot-number changed state to RUNNING
```

4 Verify the card by entering **slots**:

```
zSH> slots
Uplinks
  a:*MXK EIGHT GIGE (RUNNING)
  b: MXK EIGHT GIGE (RUNNING)
Cards
  1: MXK 8 PORT GPON (RUNNING)
 15:*MTAC RING (RUNNING)
```

An asterisk (*) indicates the card is an active card.

5 Verify the **card-profile** for the MTAC card:

```
zSH> get card-profile 1/15/5003 shelf/slot/type
Please provide the following: [q]uit.
sw-file-name: -----> {mtacring.bin}:
admin-status: -----> {operational}:
upgrade-sw-file-name: ---> {}:
upgrade-vers: -----> {}:
admin-status-enable: ----> {enable}:
sw-upgrade-admin: -----> {reloadcurrrev}:
sw-enable: -----> {true}:
sw-upgrade-enable: -----> {false}:
card-group-id: -----> {2}:
hold-active: -----> {false}:
weight: -----> {nopreference}:
card-line-type: -----> {ds1}:
card-atm-configuration: -> {notapplicable}:
card-line-voltage: -----> {not-used}:
maxvpi-maxvci:-----> {notapplicable}:
card-init-string:-----> {}:
wetting-current:-----> {disabled}:
pwe-timing-mode:-----> {none}:
```

6 View card information including the state of the card and how long the card has been running:

```
zSH> slots 15
Type           : *MTAC RING
Card Version   : 00001
EEPROM Version : 2
Serial #       : 114079
CLEI Code      : No CLEI
Card-Profile ID : 1/15/5003
Shelf          : 1
Slot           : 16
```



```

ROM Version      : MALC CAN 1.7.0.11
Software Version: MXK 1.16.2.127
State           : RUNNING
Mode            : FUNCTIONAL
Heartbeat check : enabled
Longest hbeat   : 103
Fault reset     : enabled
Uptime          : 7 days, 5 hours, 36 minutes

```

Viewing active redundant cards

You can use the **showactivecards** command to view all active cards in the system that are part of a redundant card group:

```

zSH> showactivecards
Shelf/Slot  Group Id  Card Type
-----
1:  1/15 2          MTAC RING
2:  1/a  1          MXK EIGHT GIGE

```

View additional card information

View the EPROM version of the card:

```

zSH> eeshow card 15
EEPROM contents: for slot 15
EEPROM_ID   : 00 -- CARD
Version     : 02
Size        : 054
CardType    : 05003 -- MTAC_RING
CardVersion : 00001
SerialNum   : 00114079
ShelfNumber : 00001
CLEI Code   : No CLEI
Cksum       : 0x0A66

```

Perform line testing using MTAC cards with external testing set

This section describes how the MTAC family of cards support external line testing:

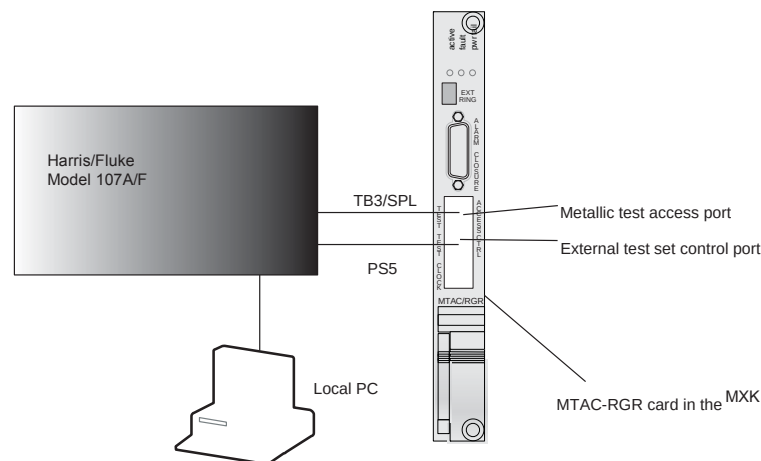
- [Connect the external test set to the MTAC card, page714](#)
- [Connect the test measurement device to the metallic test access port, page715](#)
- [Connect a console to the external test set control port, page717](#)

Connect the external test set to the MTAC card

The external test set is connected to the MTAC card through the metallic test access port and the external test set control port. The following figure details how an external test set can be connected to the MTAC card. (external test sets are also known as external test heads, external test units, and remote test units.)

The MXK enables connecting the external test set to the MTAC card to set test relays. The default baud rate is 9600 bps. (This can be changed by modifying the **rs232-profile**.)

Figure 59: External test set connected to a MTAC card



For example, to test the integrity of a line by Harris external test set, issue the **test aid** command, using the shelf, slot, and port, as a numeric keyword. For shelf 1/slot 5/port 1, issue the command **test aid=1-5-1**. Sample output is provided below.

```
HARRIS>test aid=1-5-1
DN: PAIR: SITE: TEST CHAN: 07/18/2006 15:00
NLT: PASS LDT: N/A NPA: 910 CO: CLLI:OKLAND
AID: 1-5-1 ACC:TRUNK-WB COND: OUTWARD TTYPE: LOOP SUFF:
DC SIGNATURE AC SIGNATURE NOISE
KOHMS VOLTS KOHMS VOLTS CPE CAP 60HzINDUCED
C-MESSAGEdBrcC
9999 0.00 9999 0.00 NO 0.00 T-R 37.5 TO GROUND
9999 0.00 9999 0.00 NO 0.00 T-G .002 mA T-g 0.00 METALLIC
9999 0.00 9999 0.00 NO 0.00 R-G .002 mA R-G NOISE BAL
          0.00 Mutual ( ) NOISE
UNBALANCE: 0.00% TIP LENGTH: .001 KF HIST VER: K UP, K DN
+-----+-+ +-+
| DLC |M| |M| CABLE +--+ +--+
```

```
| |a|=|D|=====| DP |====| CPE |
|DSLAM|T| |F| +--+ +--+
+-----+--+ +--+
VER35: OPEN IN EQUIPMENT
Dispatch: OFFICE (No CPE Seen)
```

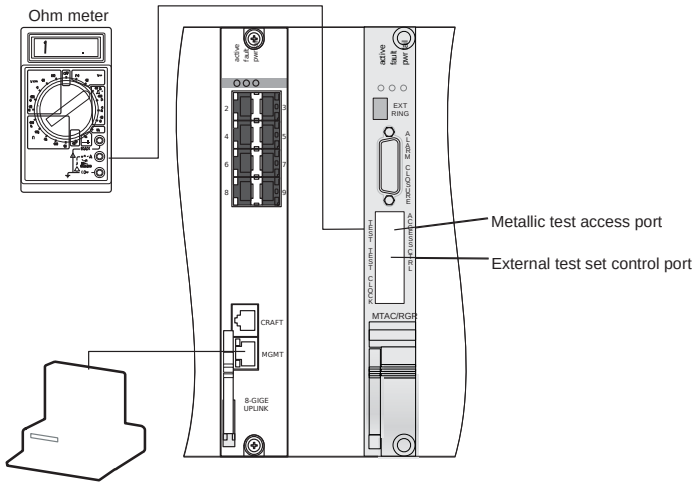
☐ **Note:** Refer to various external test set user guides for detail.

☐ **Note:** Only the pair of **Test out tip 1** and **Test out ring 1** is available to be used for loop testing.

Connect the test measurement device to the metallic test access port

If the user wants to manually measure the line integrity, the user can connect the metallic test access port on the MTAC card with a manual test measurement device, such as Ohm meter or voltage meter.

Figure 60: Manual test measurement device connected to a MTAC card



The MXK creates **mtac-profile** for each card installed in the system for manually changing test modes. After connecting the manual test measurement device, use the **mtac-linetest** command to set the relay options.

The following table describes the supported parameters in the **mtac-profile**.

Table 92: mtac-profile command parameters description

Parameter	Description
ifIndex	Specifies the ifindex of the physical line to be tested. If no line is being tested, this value is 0. Values: A physical interface on the system. In the format <i>shelf/slot/port/subport/type</i> Default: 0 This parameter cannot be modified while a test is in progress. The ability of a physical line to support a metallic test may vary depending on the cards installed and the external test equipment.
test_mode	Specifies metallic test mode for a given line. The test mode can be changed only if the ifIndex parameter is set to a nonzero value. Values: mtacModeLookOut The MXK service port is disconnected and the subscriber line is metalically routed to the MTAC metallic test access port. This allows the testing of line with or without a subscriber terminal. mtacModeNone No MTAC test is in progress. Default: mtacModeNone

The following example enables a manual test measurement device to access to the ADSL interface on *shelf 1, slot 3, port 1*:

```

zSH> update mtac-profile 1
Please provide the following: [q]uit.
ifIndex: ---> {0/0/0/0/0} 1/3/1/0/adsl
test_mode: -> {mtacmodenone} mtacmodelookout
Bad enum value 0: field test_id
test_id: ---> {NONE(0)}:
param1: ----> {0}:
param2: ----> {0}:
param3: ----> {0}:
param4: ----> {0}:
param5: ----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

To stop access to the interface, set the **mtac-profile** back to the defaults:

```

zSH> update mtac-profile 1

mtac-profile 1
Please provide the following: [q]uit.
ifIndex: ---> {1/3/1/0/adsl} 0/0/0/0/0
test_mode: -> {mtacmodelookout} mtacmodenone
Bad enum value 0: field test_id
test_id: ---> {NONE(0)}:
param1: ----> {0}:

```

```

param2: ----> {0}:
param3: ----> {0}:
param4: ----> {0}:
param5: ----> {0}:
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.

```

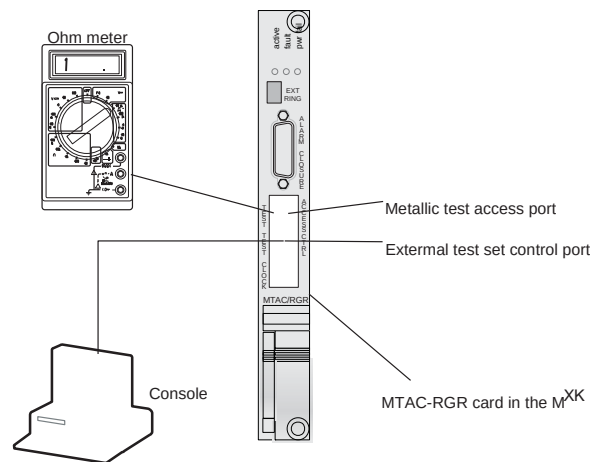


Note: The **mtac-profile** must be set back to its defaults before a line can be specified for test access.

Connect a console to the external test set control port

The user also can connect the external test set control port on the MTAC card with a console to input commands. The metallic test access port on the MTAC card would be connected with a manual test measurement device, such as Ohm meter or voltage meter to read the test results.

Figure 61: Console connected to a MTAC card



Note: These commands are used on the MTAC card external test set control port, not on the MXK uplink card zhong shell.

Use the MTAC external test set control port commands to determine what the state of the card is, either in Idle or Test mode, and to determine whether the line test has been successful.

The MTAC external test set control port commands are:

> **mtac-status**

> **mtac-linetest portaddr mode [linetype] [force]**

Note that the force parameter can only performs on voicefxs lines.

```

> mtac-status
Relay 1 in idle mode.
> mtac-linetest 1/13/1 lookout
Successful - In TestMode
> mtac-status
Relay 1 in lookout mode. Used by 1/13/1 iftype=102
> mtac-linetest 1/13/1 release
Successful - Returned to operational state
> mtac-status
Relay 1 in idle mode
> mtac-linetest 1/13/1 lookout adsl
Successful - In TestMode
> mtac-status
Relay 1 in lookout mode. Used by 1/13/1 iftype=94
> mtac-linetest 1/13/1 release
Successful - Returned to operational state

```

Perform internal line testing with MTAC/RING-ENH card

The MXK-MTAC/RING-ENH card is able to perform Metallic Loop Testing (MLT) without an external test set.

The MXK-MTAC/RING-ENH card comes with an integrated test head, thus, each shelf has its own dedicated test head. This means user can now communicate with each shelf concurrently to perform test out sessions.

This section describes the following information:

- [Working with the MTAC line test command, page718](#)
- [Test IDs, page720](#)
- [Metallic loop tests, page722](#)
- [Troubleshooting with metallic loop tests, page738](#)
- [Auto-calibration, page741](#)
- [Lookout block diagram, page742](#)

Working with the MTAC line test command

Use **mtac-linetest** commands to specify the test mode (lookout) and other test parameters for the internal line test.

The **mtac-linetest** command syntax is:

mtac-linetest portaddr mode testid [linetype] [force]

The **mtac-linetest** command has the required components of port address, mode, and test identifier; the optional components of linetype and parameter force.

```

zSH> mtac-linetest
Valid options for testid:

```

```

none abort foreigndcvoltage foreignacvoltage
dcloopresistance 3elementr      esistance
3elementcapacitance receiveroffhook distancetoopen
foreignacc      urrents ringerequiv
dtmfandpulsedigitmeasurement noisemeasurement toneg
eneration transhybridloss drawandbreakdialtone
dcfeedselftest onandoffh      ookmeasurement
meteringselftest transmissionselftest howlertest readloo
pandbatteryconditions
Valid parameters for testid:
    tonegeneration: p1=duration[sec] p2=freq[hz]
Usage: mtac-linetest <portaddr> <mode> <testid>
[<linetype>] [force] [p1] [p2] [p3] [p4] [p5]
Description:
    Execute tac test
Arguments:
    <portaddr> - port address in shelf/slot/port
    <mode> - lookout, lookin, release, bridge
    <testid> - A builtin tac linetest
    <linetype> - adsl, ds1, shdsl, isdn, vdsl, voiceebs and
voicefxs. De      fault voicefxs.
    force - override option regardless if line is in use
    p(1-5) - optional parameters for individual line test

```

[Table93](#) lists supported parameters in the **mtac-linetest** command.

Table 93: mtac-linetest command parameters description

Parameter	Description
portaddr	Specifies the port address of the physical line to be tested. Values: A port address on the system. In the format <i>shelf/slot/port</i>
mode	Specifies metallic test mode for a given line. The test mode can be changed only if the port address parameter is set to a nonzero value. Values: Lookout The MXK service port is disconnected and the subscriber line is metallically routed to the MTAC metallic test access port. This allows the testing of line with or without a subscriber terminal. Release Terminate the MTAC test that in progress. Lookin and Bridge are not supported in current version. Default: Release

Table 93: mtac linetest command parameters description (Continued)

Parameter	Description
testid	Specifies the supported line tests. Values: none, abort, foreigndcvoltage, foreignacvoltage, dcloopresistance, 3elementresistance, 3elementcapacitance, receiveroffhook, distancetoopen, foreignaccurrents, ringerequiv, dtmfandpulsedigitmeasurement, noisemeasurement, tonegeneration, transhybridloss, drawandbreakdialtone, dcfeedselftest, onandoffhookmeasurement, ringingselftest, ringingmonitor, meteringselftest, transmissionselftest, howlertest, readloopandbatteryconditions Refer to Table94 on page 720 for the detail description for each value.
linetype	Specifies the line connect type to an equipment port class. This parameter is optional. Values: adsl ds1 shdsl isdn vdsl voiceebs voicefxs Default: By default, the line type is voicefxs for POTS loops, and should be changed to the correct type when testing a loop other than POTS. Note that, the line type is voicefxs for Combo lines, not adsl.
force	This option allows seizure of a line that may be in use. Using the embedded testing is invasive to the line and should not be used for a line in use. If a line is in use and must be tested, the force option will override the current usage. Values: force
p(1-5)	Specifies the optional parameters for individual line test.

Test IDs

[Table94](#) lists the detailed description of the internal line tests that supported by MTAC/RING-ENH card.

Table 94: MTAC/RING-ENH Internal Line Tests

Test ID	Description
3elementcapacitance	This test measures tip-to-ground (T-G), ring-to-ground (R-G), and tip-to-ring (T-R) capacitance and impedance.

Table 94: MTAC/RING-ENH Internal Line Tests (Continued)

Test ID	Description
3elementresistance	This test measures tip-to-ground (T-G), ring-to-ground (R-G), and tip-to-ring (T-R) resistance.
abort	Terminate the running test.
dcfeedsselftest	This procedure verifies that the test hardware can drive currents into a load and measure the voltage across a load.
dcloopresistance	This test measures DC loop resistance using one of the following algorithms: Forward/Reverse Polarity or Offset Compensation.
distancetoopen	This test estimates the distance to an open-circuit by analyzing the results of a 3 elements resistance test and a 3 elements capacitance test.
drawandbreakdialtone	This test verifies the capability of the line circuit to detect off-hook and on-hook, the communication channel to the switching center, and the voice path from the switching center. This test is performed with the call-processing function enabled on the line under test. Note that this test will be supported in the future release.
dtmfandpulsedigit measurement	This test detects and measures a DTMF digit, pulse digit, or hook-switch flash. Only one digit or flash is reported for each invocation of this test. By default, a single tone is output on the line during this test.
foreignaccurrents	This test measures foreign AC currents.
foreigndcvoltage	This test examines the loop for the existence of DC voltage leaking into a line from an external source.
foreignacvoltage	The foreign AC voltage test is examining the loop for the existence of AC voltage leaking onto a line from an external source.
howlertest	This procedure generates a Howler (Receiver Off-Hook) tone until the phone goes on-hook or a timeout condition is detected.
meteringselftest	This procedure verifies that the line card can generate a metering pulse. It drives a metering signal into both a resistive load and an open-circuit using the current Metering Profile applied to the line.
none	No test. May be used when changing test modes.
nosiemeasurement	This procedure performs an active or passive noise test. Various filters may be applied to the received signal during this test. The application can apply special AC transmission coefficients during this test if desired.
onandoffhook measurement	This procedure verifies that the line circuit can detect on-hook and off-hook events.
readloopandbattery conditions	This procedure measures the instantaneous loop resistance, loop currents, and loop and battery voltages. No filtering is done during the measurement, so the results may fluctuate from one reading to the next in the presence of AC induction on the line.

Table 94: MTAC/RING-ENH Internal Line Tests (Continued)

Test ID	Description
receiveroffhook	This test determines whether the receiver is off-hook by running the DC Loop Resistance Test twice with different test currents and analyzing the results.
ringerequiv	This test calculates the Ringer Equivalency Number (REN) for the telephone attached to the line. The test supports both the regular and electronic phone REN measurement techniques.
ringingselftest	This procedure verifies that the line circuit can generate high level differential signals such as those used during line testing or application of internally generated ringing to the loop. It generates a sinusoidal waveform with the requested amplitude and drives this signal into a test load of known resistance.
ringingmonitor	This test is useful in checking the external ringing voltage given the loop cannot be disconnected while applying ringing and the ringing signal voltage cannot be reduced. This test is expected to be called on a line that has a terminating call (thus the need for applying ringing). This test uses about 3 cycles of the ringing waveform to carry out the test and then places the line to ringing state. Thus, a test is complete and we have placed ringing on the line as well to terminate the call. Please note that no ring trip would be detected during the first three cycles of the ringing signal.
tonegeneration	This test generates up to four sinusoidal tones simultaneously.
transhybridloss	This test measures trans-hybrid loss by generating a tone and measuring the reflected signal.
transmissionselftest	This procedure verifies that the line card can pass signals in the digital to analog and analog to digital directions. It measures trans-hybrid loss with open-circuit and a load impedance applied to the line. These trans-hybrid loss results are checked against expected values to generate a pass/fail result.

Metallic loop tests

This section outlines supported metallic loop tests, and provide some suggested boundary conditions as they are relevant to loop qualification:

- [3 elements capacitance test, page723](#)
- [3 elements insulation resistance test, page724](#)
- [DC feed self-test, page726](#)
- [DC loop resistance test, page726](#)
- [Distance to open test, page727](#)
- [DTMF and pulse digit measurement test, page728](#)
- [Foreign AC currents test, page729](#)

- [Foreign DC voltage test, page730](#)
- [Foreign AC voltage test, page730](#)
- [Howler test, page731](#)
- [Metering self test, page732](#)
- [Noise test, page732](#)
- [On-Off hook transition test, page733](#)
- [Loop and battery condition test, page733](#)
- [Receiver off-hook test, page734](#)
- [Ringer equivalency number test, page735](#)
- [Ringing self test, page735](#)
- [Ringing monitor test, page736](#)
- [Tone generation test, page736](#)
- [Trans-hybrid loss test, page737](#)
- [Transmission self test, page738](#)



Note: All the tests have the test time information as Time Started and Time Ended. The number listed in the Time Started and Time Ended are in hundredth of a second resolution. A typical test takes about 1.5 to 2 seconds.

3 elements capacitance test

The 3 elements capacitance test measures tip-to-ground (T-G), ring-to-ground (R-G), tip-to-ring (T-R) capacitance, and impedance.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/3/42 lookout 3elementcapacitance
Successful - In TestMode
Time Started: 334096
Time Ended: 334633
Three-Element capacitance Results
      (T-G)CAPACITANCE=      217.67 NFARADS
      (R-G)CAPACITANCE=      217.51 NFARADS
      (T-R)CAPACITANCE=      397.66 NFARADS
      (T-G)55Hz AC IMPEDANCE= 13.01 KOHMS
      (R-G)55Hz AC IMPEDANCE= 13.09 KOHMS
      (T-R)55Hz AC IMPEDANCE= 7.23 KOHMS
-----
Successful - Returned to operational state
zSH>
```

3 elements capacitance test result description:

- (T-G) CAPACITANCE, (R-G) CAPACITANCE, (T-R) CAPACITANCE:

- Reports the tip-to-ground, ring-to-ground, and tip-to-ring capacitances in nanofarads respectively. "NOT MEASURED" means the capacitance cannot be measured.
- (T-R) CAPACITANCE value can be used to indicate whether there is a phone attached. In most the case, a capacitance less than 60 nF indicates there is no load. A value greater than 60 nF indicates there is a load attached, possibly a phone set.

NOTE: a modern phone with electronic ringer may have less than 60 nF between its Tip and Ring.

A "NOT-MEASURED" value in (T-R) CAPACITANCE may indicate the phone is off-hook. In this case, run the 3 element resistance test to verify the resistance value between Tip and Ring.

- (T-G) 55Hz AC IMPEDANCE, (R-G) 55Hz AC IMPEDANCE, 55Hz (T-R) AC IMPEDANCE:

Reports the tip-to-ground impedance at 55Hz in kOhms. "NOT MEASURED" means the impedance cannot be measured. If the result is less than 1200 kOhms, the actual measured value is displayed as the floating-point number. Otherwise, ">1200 KOHMS (OPEN)" is displayed.

3 elements insulation resistance test

The 3 elements insulation resistance test measures the resistance of an open loop. These measurements include resistance tip-to-ground (T-G), ring-to-ground (R-G), tip-to-ring (T-R); foreign DC voltage (T-G) and (R-G); and foreign DC current in the tip and ring leads.

Note that this test measures conditions on an open loop (T-R resistance is high). If the loop is closed, or the T-R resistance is under 2 kOhms, the DC loop resistance test should be used for a more accurate measurement.

Note that, the foreign DC voltage results from this test are not as accurate as those returned by the individual foreign DC voltage Test, but the overall testing time may be reduced by using results of this test instead of additionally running the foreign DC voltage test.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/17/4 lookout 3elementresistance
Successful - In TestMode
Time Started: 1129209
Time Ended: 1129456
Three-Element Resistance Results
      (T-G) DC RESISTANCE= > 1200      KOHMS(OPEN DC )
      (R-G) DC RESISTANCE= > 1200      KOHMS(OPEN DC )
      (T-R) DC RESISTANCE= 946.68      KOHMS
      (T-G) FOREIGN DC VOLTAGE= NONE    VOLTS
      (R-G) FOREIGN DC VOLTAGE= NONE    VOLTS
      TIP FOREIGN DC CURRENT= 0.00      MILLIAMPS
      RING FOREIGN DC CURRENT= 0.00      MILLIAMPS
```

 Successful - Returned to operational state
 ZSH>

3 elements resistance test result description:

- (T-G) DC RESISTANCE, (R-G) DC RESISTANCE, (T-R) DC RESISTANCE:
 - If the resistance is less than 150 ohms, it is considered to be very small, and interpreted as a short circuit or fault. Use DC loop resistance test for T-R resistance under 2 kOhms.
 - If the result could not be calculated because of some fault, NOT MEASURED is printed.
 - If the resistance is larger than 1200 kOhms, it is considered to be too high to be measured accurately, and interpreted as open-circuit.
 - Otherwise, the resistance result is considered as normal, and interpreted as a floating-point number.
- (T-G) FOREIGN DC VOLTAGE, (R-G) FOREIGN DC VOLTAGE:
 - If the result is less than 5 V, it is considered to be a normal value, and the foreign DC voltage is printed as a floating-point number.
 - If the result is between 5 V to 12 V, it is considered to be marginally normal, and printed as a floating-point number.
 - If the result is greater than 12 V, it is considered as not a normal value, and should be investigated.
 - If the result is printed as NONE, it means the result is normal for loop start, data loops, and CPE.
- TIP FOREIGN DC CURRENT, RING FOREIGN DC CURRENT:
 - If the result is less than 1 milliamps (mA), it is considered to be a normal value, and the tip foreign DC current is printed as a floating-point number.
 - If the result is between 1 mA to 3 mA, it is considered to be marginally normal, and printed as a floating-point number.
 - If the result is greater than 3 mA, it is considered as not a normal value, and should be investigated. If the result is greater than 8 mA, it is printed as "> 80 MILLIAMPS", if the result is between 3 mA to 8 mA, the result is printed as floating-point number.
 - If the result is printed as NONE, it means the result is normal for loop start, data lines loops, and CPE.

DC feed self-test

This self test puts a 0.89 kOhms test load on the line, and measures the return in order to determine if appropriate levels are available on the line.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout dcfeedselftest
```

```
Successful - In TestMode
```

```
Time Started: 9023093
```

```
Time Ended: 9023383
```

DC Feed Self-Test Results

TEST PASSED=	Yes	
MEASURED TEST LOAD=	892.34	OHMS
MEASURED HIGH BAT VOLTAGE=	-50.42	VOLTS
(T-R) MEASURED VOLTAGE=	40.87	VOLTS
CURRENT IN TEST LOAD (BAT=High, POL=Normal)=	16.21	MILLIAMPS
CURRENT IN TEST LOAD (BAT=High, POL=Reverse)=	16.30	MILLIAMPS
CURRENT IN TEST LOAD (BAT=Low, POL=Normal)=	15.69	MILLIAMPS

```
-----  
Successful - Returned to operational state
```

DC feed self test result description:

- TEST PASSED indicates whether the test passed. It based solely on the measured test load, high battery potential and tip-ring voltage.
- MEASURED TEST LOAD reports the measured test load resistance. If this result is not within 10% of the nominal load resistance (0.89 kOhms), then TEST PASSED is set to NO.
- MEASURED HIGH BAT VOLTAGE reports the measured high battery voltage. It is nominal at -48 VDC. Acceptable ranges for this value are -42 to -56 VDC.
- (T-R) MEASURED VOLTAGE reports the measured tip-ring voltage while high current is driven. If the magnitude of the tip-ring voltage plus 2.66 V SLIC head-room plus $(90.4 / \text{MEASURED TEST LOAD}) \times (\text{T-R})$ MEASURED VOLTAGE is not within the MEASURED HIGH BAT VOLTAGE $\pm (1.1 \text{ V plus } 10\% \text{ of the magnitude of the high battery voltage})$, then TEST PASSED is set to NO.

DC loop resistance test

The DC loop resistance test measures the resistance on a line, longitudinal imbalances, and other characteristics. This test is useful for low loop resistance, generally less than 4 kOhms. For higher resistance loop the 3 elements resistance test is more accurate.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout dcloopresistance
```

```
Successful - In TestMode
```

```
Time Started: 9025472
```

```
Time Ended: 9025648
```

DC loop resistance Test Results

LOOP RESISTANCE=	3.95KOHMS
COMMON MODE CURRENT Phase 1=	0.00 MILLIAMPS

```
COMMON MODE CURRENT Phase 2=      0.00 MILLIAMPS
Voltage Saturation=                Yes
COMMON MODE CURRENT Degradation=No
```

```
-----
Successful - Returned to operational state
```

DC loop resistance test result description:

- LOOP RESISTANCE reports the measured loop resistance in kOhms.
- COMMON MODE CURRENT Phase 1 reports the common mode current measured during the test first phase.
- COMMON MODE CURRENT Phase 2 reports the common mode current measured during the test second phase.
- Voltage Saturation
 - = Yes indicates that the tip-ring voltage approached the battery voltage while attempting to drive the requested test current through the loop. The users should run the 3 element resistance test to get a more accurate measurement.
 - = No is a normal measurement.
- COMMON MODE CURRENT Degradation.
 - = Yes indicates that the test results may be inaccurate due to excessive common-mode current. The users should run the 3 element resistance test to get a more accurate measurement.
 - = No is a normal measurement.

Distance to open test

This test estimates the distance to an open-circuit by analyzing the results of a 3 elements resistance test and a 3 elements capacitance test.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/3/42 lookout distancetoopen
Successful - In TestMode
Time Started: 626510
Time Ended:   627395
Distance to open Results
      Distance to open=      4379.50      Meters
      Capacitence(measured)= 218.97      NFARADS
-----
Successful - Returned to operational state
zSH>
```

Distance to open test result description:

- Distance to open reports the estimated distance to an open-circuit in meters.

- Capacitance (measured) reports the measured capacitance value in nanofarads.
- If the test fails, one or both of the following errors will be displayed:
 - Test failed because the 3eleResistance failed – the Three-Element Insulation Resistance Test results show excessive current leakage.
 - Test failed because the 3eleCapcitence failed – the Three-Element Capacitance Test could not accurately measure the tip-to-ground and ring-to-ground capacitance.

DTMF and pulse digit measurement test

This test detects and measures a DTMF digit, pulse digit, or hook-switch flash. Only one digit or flash is reported for each invocation of this test. By default, a single tone is output on the line during this test. The test runs for approximately 4 seconds.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout
dtmfandpulsedigitmeasurement
Successful - In TestMode
Time Started: 9032539
Time Ended: 9032966
```

```
DTMF/pulse Results
DTMF/pulse test timed out
```

```
-----
Successful - Returned to operational state
```

DTMF and pulse digit measurement test result description:

- If no DTMF digits detected, the test result prints “DTMF/pulse test timed out”.
- If a DTMF digit was detected, the test result prints the real measurement as floating-point number.
- If a DTMF digit was detected and it has time to do a fourier transform the test result prints:
 - DTMF DIGIT <digit, e.g. 4>
 - DTMF SAMPLE SIZE= <number of FFT samples taken>
 - DTMF FIRST TONE= <frequency, e.g. 800> Hz
 - DTMF FIRST TONE LEVEL= <amplitude, e.g. 12> dBm
 - DTMF SECOND TONE= <frequency, e.g. 800> Hz
 - DTMF SECOND TONE LEVEL= <amplitude, e.g. 12> dBm
- If a Tone was detected but no DTMF digit detected and it has time to do a fourier transform it prints:

- DTMF DIGIT NO DIGIT DECODED
- DTMF SAMPLE SIZE= <number of FFT samples taken>
- DTMF FIRST TONE= <frequency, e.g. 800> Hz
- DTMF FIRST TONE LEVEL= <amplitude, e.g. 12> dBm
- DTMF SECOND TONE= <frequency, e.g. 800> Hz
- DTMF SECOND TONE LEVEL= <amplitude, e.g. 12> dBm
- If a pulse digit is detected it prints:
 - PULSE DIGIT= <digit, e.g. 7>
 - PULSE MINBRK= <% number> % OF AVG PERIOD
 - PULSE MAXBRK= <% number> % OF AVG PERIOD
 - PULSE AVGBRK= <% number> % OF AVG PERIOD
 - PULSE RATE= <number> PER SEC

(The MINBRK, MAXBRK, AVGBRK are a percentage of the average pulse period indicated by pulse rate.)
- If a hook flash is detected it prints “PULSE INTERVAL= <number e.g. 1120> mSEC”.
- If a disconnect is detected it prints “DISCONNECT DETECTED”.

Foreign AC currents test

This test measures foreign AC currents.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout foreignaccurents
Successful - In TestMode
Time Started: 9035116
Time Ended: 9035207

Foreign AC current Results
TIP FOREIGN AC CURRENT= NONE MILLIAMPS
RING FOREIGN AC CURRENT= NONE MILLIAMPS
-----
Successful - Returned to operational state
```

The foreign AC currents test result description:

- TIP FOREIGN CURRENT reports the measured tip lead current in milliamps.
- RING FOREIGN CURRENT reports the measured ring lead current in milliamps.

Foreign DC voltage test

The foreign DC voltage test is examining the loop for the existence of DC voltage leaking onto a line from an external source.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout foreigndcvoltage
Successful - In TestMode
Time Started: 9036757
Time Ended: 9036966
```

Foreign DC Voltage Test Results

```
Test Passed=Yes
(T-G)FOREIGN DC VOLTAGE= 0.02 VOLTS(OK)
(R-G)FOREIGN DC VOLTAGE= 0.00 VOLTS(OK)
(T-R)FOREIGN DC VOLTAGE= 0.02 VOLTS(OK)
```

Successful - Returned to operational state

The foreign DC voltage test result description:

(T-G) FOREIGN DC VOLTAGE, (R-G) FOREIGN DC VOLTAGE, (T-R) FOREIGN DC VOLTAGE:

- <3 volts is normal. (OK) is printed after the measured data.
- >6 volts indicates a fault, it need to retest and follow-up. (FAULT) is printed after the measured data.
- = or > 100 volts indicates the presence of hazardous levels, and should be considered a dangerous fault. (HAZARDOUS) is printed after the measured data.
- For lines using ADSL2+, the voltage level for tip to ground should be less than 3 volts to ensure a stable DSL connection.

Foreign AC voltage test

The foreign AC voltage test is examining the loop for the existence of AC voltage leaking onto a line from an external source.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout foreigndcvoltage
Successful - In TestMode
Time Started: 9038284
Time Ended: 9038424
```

Foreign AC Voltage Test Results

```
(T-G)FOREIGN AC VOLTAGE= 0.00 VRMS(NONE)
(R-G)FOREIGN AC VOLTAGE= 0.00 VRMS(NONE)
(T-R)FOREIGN AC VOLTAGE= 0.00 VRMS(NONE)
```

Successful - Returned to operational state

Foreign AC Voltage test result description:

- (T-G) FOREIGN AC VOLTAGE, (R-G) FOREIGN AC VOLTAGE:
 - < 3 AC volts rms (Vrms), (NONE) will be printed out after a real measurement.
 - = or > 3 to = or < 10 AC Vrms is a normal and good measurement. It is normal for loop start, data lines loops, and CPE. (OK) will be printed out after a real measurement.
 - >10 to = or < 50 AC Vrms is a fault. It should be retested and then investigated. (FAULT) will be printed out after a real measurement.
 - >50 AC Vrms indicates the presence of hazardous levels, and should be considered a dangerous fault. (HAZARDOUS) will be printed out after a real measurement.
 - For lines using ADSL2+, the voltage levels for tip-to-ground and ring-to-ground should be less than 10 volts to ensure a stable DSL connection.
- (T-R) Foreign AC VOLTAGE:
 - < 3 Vrms is a normal and good measurement. It is normal for loop start, data lines loops, and CPE. (NONE) will be printed out after a real measurement.
 - =or >3 to = or < 5 Vrms is a fault. It should be retested and then investigated. (FAULT) will be printed out after a real measurement.
 - >50 Vrms indicates the presence of hazardous levels, and should be considered a dangerous fault. (HAZARDOUS) will be printed out after a real measurement.
 - For lines using ADSL2+, the voltage level for tip to ring should be less than 3 volts to ensure a stable DSL connection.

Howler test

This procedure generates a Howler (Receiver Off-Hook) tone until the phone goes on-hook or a timeout condition is detected.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout howlertest
Successful - In TestMode
Time Started: 9039942
Time Ended: 9040152
Howler Test results
Running US Howler Test
-----
Successful - Returned to operational state
```

The howler test result description:

Depending on the system profile, the howler test prints “Running US Howler Test”, “Running Australian Howler Test”, or “Running UK Howler Test”. If the system profile cannot be read, the test prints “Failed to access the system profile”, and stop the test.

Metering self test

This procedure verifies that the line card can generate a metering pulse. It drives a metering signal into both a resistive load and an open-circuit using the current Metering Profile applied to the line.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/3/42 lookout meteringselftest
Successful - In TestMode
Time Started: 396242
Time Ended: 396298
Metering Self-Test Results
      TEST PASSED=                               Yes
      Peak metering Voltage Resistive Load= 1.22    VOLTS
      Peak metering Voltage Open circuit= 1.40    VOLTS
-----
Successful - Returned to operational state
zSH>
```

The metering self test result description:

- Peak metering Voltage Resistive Load reports the peak voltage of the metering signal with the circuit connected to a resistive load.
- Peak metering Voltage Open Circuit reports the peak voltage of the metering signal with the circuit open.

Noise test

The noise test measures the amount of noise in dBm on the line, relative to TLP 0. This provides measurements in dBm0 units.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout noisemeasurement
Successful - In TestMode
Time Started: 9047559
Time Ended: 9047703

Noise Test results
NOISE= -67.23 dBm0
-----
Successful - Returned to operational state
```

Noise test result description:

- Noise below -45 dBmO is an average loop (LSB switching noise approaches -45 dBmO).

- Noise between -44 and -10 dBmO is too noisy and should be retested and investigated.

On-Off hook transition test

This on-hook to off-hook test allows the MLT to determine if a loop can successfully complete a simulated hook state transition.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout onandoffhookmeasurement
Successful - In TestMode
Time Started: 9049153
Time Ended: 9049231
```

```
ON-OFF Hook Self-Test Results
PASSED
```

```
-----
Successful - Returned to operational state
```

The on-off hook transition test result description:

- PASSED indicates that the test was passed.
- ABORTED indicates that the line was off-hook when the test was started.
- HW_FAULT indicates that the test failed because the line circuit did not properly detect on-hook and off-hook state changes.
- UNKNOWN indicates some unexpected error occurred.

Loop and battery condition test

The loop and battery condition test measures the instantaneous loop resistance, loop currents, and loop and battery voltages. No filtering is done during the measurement, so the results may fluctuate from one reading to the next in the presence of AC induction on the line.

The following example tests the POTS line on shelf 1, slot 4, port 1 with a forced readloopandbatteryconditions test using lookout mode, and provides the outputs:

```
zSH> mtac-linetest 1/4/1 lookout readloopandbatteryconditions force
Successful - In TestMode
Time Started: 9053736
Time Ended: 9053737
```

```
Read Loop and Battery Condition TestResults
Read Loop and Battery Condition Test
```

LOOP resistance=	not measured	
Common-mode (longitudinal)current=	not measured	
(T-R) (metallic) current=	not measured	
(T-R) voltage=	not measured	
Lowest battery voltage (measured)=	-50.30	VOLTS
Highest battery voltage (measured)=	-50.41	VOLTS

```
Positive battery voltage=          1.40          VOLTS
Metering Voltage (measured)=      0.00          VOLTS
NOTE: the metering voltage is only valid if a metering pulse is currently
being generated.
```

Successful - Returned to operational state

Loop and battery condition test result description:

- Loop resistance reports the loop resistance in kOhms. If the loop resistance cannot be measured in the present line state, “not measured” is reported.
- Common-mode (longitudinal) current reports the longitudinal (common-mode) current in milliamps. If the longitudinal current cannot be measured in the present line state, “not measured” is reported.
- (T-R) (metallic) current reports the metallic (tip-to-ring) current in milliamps. If the metallic current cannot be measured in the present line state, “not measured” is reported.
- (T-R) voltage reports the tip-to-ring voltage in volts. If the tip-to-ring voltage cannot be measured in the present line state, "not measured" is reported.
- Lowest battery voltage (measured) reports the voltage of battery with the lowest absolute value in volts.
- Highest battery voltage (measured) reports the voltage of the battery with the highest absolute value in volts.
- Positive battery voltage reports the positive battery voltage in volts.
- Metering Voltage (measured) reports the peak metering signal voltage observed across tip and ring since the start of the metering pulse.

Receiver off-hook test

The receiver off-hook test allows the MLT to determine if a loop has one or more telephones that are off-hook at the far end of the circuit.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout receiveroffhook
Successful - In TestMode
Time Started: 9057457
Time Ended:    9057635
```

Receiver Off-Hook Test Results

```
OFF-HOOK=          No
RL00P out of range=  Yes
```

Successful - Returned to operational state

Receiver off-hook test results are described in the following table:

Table 95: Receiver off-hook test result description

OFF-HOOK	RLOOP out of range	Test Result Description
NO	NO	The DC Loop Resistance Test returned real resistance values, but they are not characteristic of an off-hook receiver.
YES	NO	This test measured loop resistances that suggest an off-hook receiver.
NO	YES	The DC Loop Resistance Test failed to measure loop resistance or returned an unreasonable result. This is most likely due to the receiver being on-hook.
YES	YES	This test never returns this result.

Ringer equivalency number test

This test calculates the Ringer Equivalency Number (REN) for the telephone attached to the line. The test supports both the regular and electronic phone REN measurement techniques.

The following example provides the sample command and output:

```

zSH> mtac-linetest 1/3/42 lookout ringerequiv
Successful - In TestMode
Time Started: 415643
Time Ended: 415740
Ringer Equivalence Number Test Results
      REN=                0.49                RINGEQIV
      Measured Zload=      14.26                KOHMS
      COMMON MODE CURRENT Degradation=        no
-----
Successful - Returned to operational state
zSH>

```

Ringer equivalency number test result description:

- REN reports the measured Ringer Equivalency Number (REN).
- Measured Zload reports the measured ringer impedance in kOhms and only applies to the regular phone REN test. It is set to zero if the application ran an electronic phone REN test.
- COMMON MODE CURRENT Degradation is YES if the test results may be inaccurate due to excessive common-mode current. This flag only applies to the regular phone REN test and is set to zero if the application ran an electronic phone REN test.

Ringing self test

The ringing self test is a simulation of ringing on current able to be passed on the line. As a note, no actual ringing will be audible due to low voltage used.

The following example provide the sample command and output:

```

zSH> mtac-linetest 1/4/1 lookout ringingselftest
Successful - In TestMode
Time Started: 9061926
Time Ended:    9062086

Ringing Self-Test Results
      TEST PASSED=                Yes
      RLOOP=                2.53                KOHMS
      ON HOOK TO OFF HOOK TRANSITION DETECTED= Yes
-----
Successful - Returned to operational state

```

This test is informational, and is used to determine if loop conditions on the line will allow ringing current to reach the far end of the circuit. This test should pass on valid loops.

Ringing monitor test

The ringing monitor test checks the external ringing voltage given the loop cannot be disconnected while supplying ringing and the ringing signal voltage cannot be reduced.

The following example provides the sample command and output:

```

zSH> mtac-linetest 1/4/1 lookout ringingmonitor
Successful - In TestMode
Time Started: 9078335
Time Ended:    9078393

Ringing Monitor Results
      Test Aborted due to off hook=No
      Ring Voltage=                0.00 VRMS(NONE)
-----
Successful - Returned to operational state

```

Ring monitor test result description:

- Test abort due to off hook indicates whether the test was aborted due to off-hook detection at the beginning of the test.
- Ring voltage reports the measured external ringing voltage in RMS volts.

Tone generation test

This test generates up to four sinusoidal tones simultaneously. It also provides the optional parameters to set the maximum duration and frequency of the tone. By default, the duration is 2 seconds, and frequency is 1000 Hz.

The following examples provide the sample commands, and the outputs for the succeeded tests.

A basic tone generation test:

```

zSH> mtac-linetest 1/4/1 lookout tonegeneration
Successful - In TestMode

```



```
Time Started: 9079951
Time Ended:   9080179
```

```
-----
Successful - Returned to operational state
```

A tone generation test with the maximum duration of 60 seconds.

```
zSH> mtac-linetest 1/4/1 lookout tonegeneration 60
Successful - In TestMode
Time Started: 3127104
Time Ended:   3133173
```

```
-----
Successful - Returned to operational state
```

A tone generation test with the maximum duration of 60 seconds and tone frequency of 2000 Hz.

```
zSH> mtac-linetest 1/4/1 lookout tonegeneration 180 2000
Successful - In TestMode
Time Started: 3135884
Time Ended:   3154046
```

```
-----
Successful - Returned to operational state
```

The tone generation tests in the above examples are succeed although in the output didn't show the data.

Trans-hybrid loss test

This loop test characterizes the amount of echo from a far end trans-hybrid unit. This is only found in a telephone device, and is not a valid test on a dry pair for DSL.

The following example provides the sample command and output:

```
zSH> mtac-linetest 1/4/1 lookout transhybridloss
Successful - In TestMode
Time Started: 9081809
Time Ended:   9081938
```

Transhybrid Loss Results

ECHO=	-84.13	dBm0
LOSS=	74.13	dBm0

```
-----
Successful - Returned to operational state
```

Trans-hybrid loss test result description:

- ECHO returns the measured signal echo power in dBm0.
- LOSS returns the calculated trans-hybrid loss in dB.

Transmission self test

The transmission self test attempts to determine if the line's trans-hybrid loss using a test load is greater than an allowed minimum loss. The test load value should be greater than the lower limit value. A trans-hybrid device is only found in a telephone device, and is not a valid test on a dry pair for DSL.

The following example provides the sample command and sample output:

```
zSH> mtac-linetest 1/3/42 lookout transmissionselftest
Successful - In TestMode
Time Started: 429667
Time Ended: 429743
Transmission Self Test Results
  TEST PASSED= No
  TEST ABORT, OFF_HOOK= No
  TRANS-HYBRID LOSS OPEN= 53.30 dB
  TRANS_HYBRID LOSS RLOAD= 8.98 dB
  EXPECTED TRANS-HYBRID LOSS LOWER LIMIT= 20.83 dB
  EXPECTED TRANS-HYBRID LOSS UPPER LIMIT= VERY_HIGH_THL
-----
Successful - Returned to operational state
```

This test is informational, and is used to determine trans-hybrid loss on a loop. This test should pass on valid loops. The loss on open circuits should read nominally 0. The loss on the test load should read higher than the loss lower limit. If the measured level is lower than stated lower limit, then it may indicate a problem with the line.

Troubleshooting with metallic loop tests

To diagnose the problem in the metallic loop, may takes several different MTAC tests. The following examples provide the sample troubleshooting cases.

Phone is off-hook

To troubleshoot whether the phone is off-hook, use the 3 element capacitance test and 3 element resistance test. The (T-R) CAPACITANCE value can be used to indicate whether there is a phone attached. In most cases, a capacitance less than 60 nanofarads indicates the Tip to Ring is open, there is no load (e.g. no phone attached); A value greater than 60 nanofarads indicates there is a load attached, possibly a phone set; A value "NOT MEASURED" indicates the Tip to Ring is shorted, and possibly the phone is off-hook.



Note: The following examples in this section are not using the modern phone.

A modern phone with electronic ringer may have less than 60 nanofarads between its Tip and Ring.

Here is an example of phone is off-hook (with 9600 ft cable):

- 1 At first, look the (T-R) CAPACITANCE value in the 3 element capacitance test output. A “NOT-MEASURED” value in T-R CAPACITANCE indicate the phone is possibly off-hook.

zSH> **mtac-linetest 1/7/27 lookout 3elementcapacitance force**

Three-Element capacitance Results

(T-G) CAPACITANCE=	155.09	NFARADS
(R-G) CAPACITANCE=	156.71	NFARADS
(T-R) CAPACITANCE=	NOT MEASURED	
(T-G) 55Hz AC IMPEDANCE=	16.10	KOHMS
(R-G) 55Hz AC IMPEDANCE=	15.97	KOHMS
(T-R) 55Hz AC IMPEDANCE=	NOT MEASURED	

- 2 Then run the 3 element resistance test to verify the resistance value between Tip and Ring. The “748.47 OHMS” value in (T-R) DC RESISTANCE indicates the Tip and Ring are closed or shorted. Based on this information, then we can diagnosed that the phone is off-hook.

zSH> **mtac-linetest 1/7/27 lookout 3elementresistance force**

Three-Element Resistance Results

(T-G) DC RESISTANCE=	> 1200	KOHMS (OPEN DC)
(R-G) DC RESISTANCE=	> 1200	KOHMS (OPEN DC)
(T-R) DC RESISTANCE=	748.47	OHMS
(T-G) FOREIGN DC VOLTAGE=	NONE	VOLTS
(R-G) FOREIGN DC VOLTAGE=	NONE	VOLTS
TIP FOREIGN DC CURRENT=	0.00	MILLIAMPS
RING FOREIGN DC CURRENT=	0.00	MILLIAMPS

Phone is on-hook

Here is an example of phone is on-hook (with 9600 ft 24 AWG cable):

Run the 3 element capacitance test. Look the (T-R) CAPACITANCE value in the 3 element capacitance test output. In this example, the value “124.67 NFARADS” is greater than 60 nanofarads, it indicates the phone is on-hook.

zSH> **mtac-linetest 1/7/27 lookout 3elementcapacitance force**

Three-Element capacitance Results

(T-G) CAPACITANCE=	151.11	NFARADS
(R-G) CAPACITANCE=	151.75	NFARADS
(T-R) CAPACITANCE=	124.67	NFARADS
(T-G) 55Hz AC IMPEDANCE=	16.52	KOHMS
(R-G) 55Hz AC IMPEDANCE=	16.49	KOHMS
(T-R) 55Hz AC IMPEDANCE=	20.21	KOHMS

Phone is not attached

Here is an example of no phone is attached:

Run the 3 element capacitance test. Look the (T-R) CAPACITANCE value in the 3 element capacitance test output. In this example, the value “0.79 NFARADS” is less than 60 nanofarads, it indicates the Tip to Ring is open, there is no load (e.g. no phone attached).

```
zSH> mtac-linetest 1/7/27 lookout 3elementcapacitance force
Three-Element capacitance Results
(T-G) CAPACITANCE=          1.62          NFARADS
(R-G) CAPACITANCE=          1.59          NFARADS
(T-R) CAPACITANCE=          0.79          NFARADS
(T-G) 55Hz AC IMPEDANCE=    508.99        KOHMS
(R-G) 55Hz AC IMPEDANCE=    552.75        KOHMS
(T-R) 55Hz AC IMPEDANCE=    > 1200        KOHMS (OPEN)
```

Both Tip and Ring are grounded

Here is an example where the loop line is grounded on both Tip and Ring. In this case, both 3 element resistance and 3 element capacitance tests would fail, indicating the line is shorted and grounded. With the DC loop resistance test, it may be possible to use the loop resistance value to determine the distance of the line.

- 1 At first, run the 3 element capacitance test. “NOT-MEASURED” values indicate cannot measure these values in the loop line.

```
zSH> mtac-linetest 1/7/26 lookout 3elementcapacitance force
```

```
Three-Element capacitance Results
(T-G) CAPACITANCE=          NOT MEASURED
(R-G) CAPACITANCE=          NOT MEASURED
(T-R) CAPACITANCE=          NOT MEASURED
(T-G) 55Hz AC IMPEDANCE=    NOT MEASURED
(R-G) 55Hz AC IMPEDANCE=    NOT MEASURED
(T-R) 55Hz AC IMPEDANCE=    NOT MEASURED
```

- 2 Then run the 3 element resistance test. Look the DC RESISTANCE value in the 3 element resistance test output. A “< 150” value is considered to be very small, and interpreted as a short circuit or fault.

```
zSH> mtac-linetest 1/7/26 lookout 3elementresistance force
```

```
Three-Element Resistance Results
(T-G) DC RESISTANCE= < 150          OHMS(FAULT) TEST HALTED
(R-G) DC RESISTANCE= < 150          OHMS(FAULT) TEST HALTED
```

- 3 And then run the DC loop resistance test with an 100 feet cable.

```
zSH> mtac-linetest 1/7/26 lookout dcloopresistance force
```

```
DC loop resistance Test Results
LOOP RESISTANCE=          0.07          KOHMS
COMMON MODE CURRENT Phase 1= 0.00        MILLIAMPS
COMMON MODE CURRENT Phase 2= 0.00        MILLIAMPS
```

```
Voltage Saturation=          No
COMMON MODE CURRENT Degradation= Yes
```

- 4 Or run the DC loop resistance test with a 9600 feet 24 awg cable.

```
zSH> mtac-linetest 1/7/26 lookout dcloopresistance force
```

DC loop resistance Test Results

```
LOOP RESISTANCE=          0.56          KOHMS
COMMON MODE CURRENT Phase 1= 0.00          MILLIAMPS
COMMON MODE CURRENT Phase 2= 0.00          MILLIAMPS
Voltage Saturation=          No
COMMON MODE CURRENT Degradation= Yes
```

Only Ring wire is grounded

The following example shows how to use metallic test command to diagnose the Ring to Ground (R-G) is shorted, yet, the Tip to Ground (T-G) is open.

- 1 At first, run the 3 element capacitance test. “NOT-MEASURED” values indicate cannot measure these values in the loop line.

```
zSH> mtac-linetest 1/7/26 lookout 3elementcapacitance
force
```

Three-Element capacitance Results

```
(T-G) CAPACITANCE=          NOT MEASURED
(R-G) CAPACITANCE=          NOT MEASURED
(T-R) CAPACITANCE=          NOT MEASURED
(T-G) 55Hz AC IMPEDANCE=          NOT MEASURED
(R-G) 55Hz AC IMPEDANCE=          NOT MEASURED
(T-R) 55Hz AC IMPEDANCE=          NOT MEASURED
```

- 2 Then run the 3 element resistance test. It only shows (R-G) DC RESISTANCE value, didn't show the (T-G) DC RESISTANCE value, and the value is “< 150”. This indicates Ring to Ground is shorted, Tip to Ground is open.

```
zSH> mtac-linetest 1/7/26 lookout 3elementresistance force
```

Three-Element Resistance Results

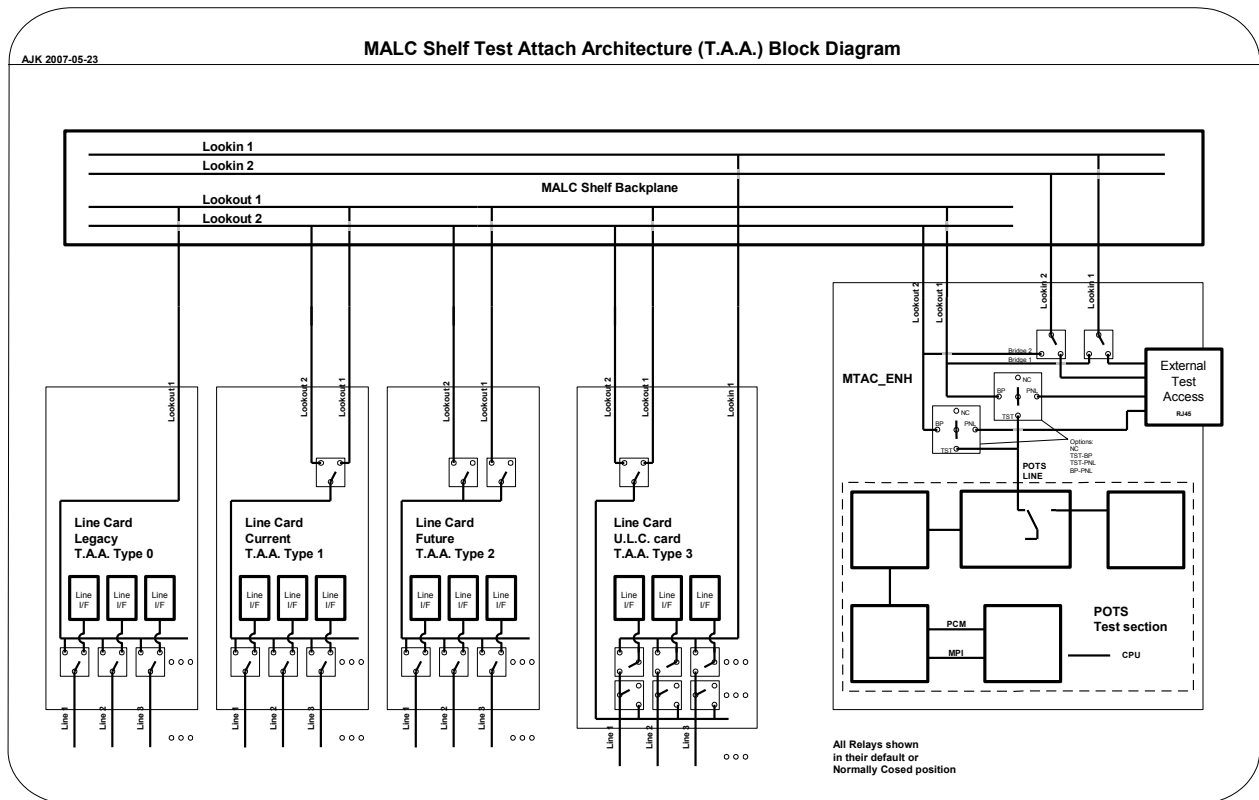
```
(R-G) DC RESISTANCE= < 150          OHMS(FAULT) TEST HALTED
```

Auto-calibration

When the **mtac-linetest** command is issued, prior to running the line test, the line card performs an auto-calibration.

Lookout block diagram

Figure 62: Lookout block diagram



Configure external alarms

The MTAC cards have a 26 pin connector that provides sensing of alarm relay contacts for up to 12 external devices. When an alarm condition occurs on the external device, the MXK sends a trap. Each of the 12 pairs of pins can be assigned to a different alarm. Use the **num2str-profile** to assign a description to an alarm relay. The description is included in traps and log messages.

The **num2str-profile** uses an index in the form:

```
shelf/slot/282/alarm-contact
```

The following example adds a description to the first alarm contact of a MTAC/RING card in *shelf* 12:

```
zSH> update num2str-profile 1/12/282/1
Please provide the following: [q]uit.
name: -> {Relay 1}: cabinet open
.....
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Configure an external clock

The MXK supports the following external clock sources:

- A recovered clock from a T1/E1 line.
- Building Integrated Timing Supply (BITS) clock.

Connecting a T1/E1 recovered clock to the MTAC card

The network T1/E1 clock on the MTAC card appears to the system as a T1/E1 interface. To connect the clock source:

- 1 Connect the T1/E1 cable to the MTAC card external clock input port which is an RJ-45 port labeled CLOCK. This is used to source the clock to the shelf using standard T1/E1 pin connections.
- 2 Configure the system to use the clock, as explained in [MXK clocking on page 98](#).

Connecting a BITS clock to the MTAC card

The external clock input port on the MTAC card uses pins 6 and 8 for ground and pin 7 for the clock reference. To connect the BITS clock (also known as 2 Mhz clock) source:

- 1 Connect the BITS clock signal to the MTAC card external clock input port which is an RJ-45 port labeled CLOCK, pin 7.
- 2 Connect the clock ground line to both pins 6 and 8 together (pin 6 is the 2 MHz input selector pin). This selects use of the 2 MHz BITS clock instead of the T1/E1 recovered clock.
- 3 Configure the **card-line-type** parameter in the MTAC **card-profile** to **e1**.

```
zSH> card add 14 linetype e1 group 2
An autogenerated card-group-id [2] is assigned for this card type.
new card-profile 1/14/5072 added, sw-file-name "tacitmring.bin", 2 options:
card-group-id 2 card-line-type e1
```

Verify the card-line-type:

```
zSH> get card-profile 1/14/5072
card-profile 1/14/5072
Please provide the following: [q]uit.
sw-file-name: -----> {tacitmring.bin}:
admin-status: -----> {operational}:
upgrade-sw-file-name: ---> {}:
upgrade-vers: -----> {}:
admin-status-enable: ----> {enable}:
sw-upgrade-admin: -----> {reloadcurrrev}:
sw-enable: -----> {true}:
sw-upgrade-enable: -----> {false}:
card-group-id: -----> {2}:
hold-active: -----> {false}:
```

```

weight: -----> {nopreference}:
card-line-type: -----> {e1}:
card-atm-configuration: -> {notapplicable}:
card-line-voltage: -----> {not-used}:
maxvpi-maxvci: -----> {notapplicable}:
card-init-string: -----> {}:
wetting-current: -----> {disabled}:
pwe-timing-mode: -----> {none}:

```

- 4 When the MTAC card is running, update **ds1-profile** by changing the **line-type** parameter to **other** and the **transmit-clock-source** to **looptiming**.

```

zSH> update ds1-profile 1-14-1-0/ds1
ds1-profile 1-14-1-0/ds1
Please provide the following: [q]uit.
line-type: -----> {e1crc}:  other
line-code: -----> {hdb3}:
send-code: -----> {sendnocode}:
circuit-id: -----> {e1}:
loopback-config: -----> {nolooop}:
signal-mode: -----> {none}:
fdl: -----> {fdlnone}:
dsx-line-length: -----> {dsx0}:
line-status_change-trap-enable: --> {enabled}:
channelization: -----> {disabled}:
ds1-mode: -----> {other}:
csu-line-length: -----> {csu00}:
clock-source-eligible: -----> {eligible}:
transmit-clock-source: -----> {throughtiming}:  looptiming
cell-scramble: -----> {true}:
coset-polynomial: -----> {true}:
protocol-emulation: -----> {network}:
signal-type: -----> {loopstart}:
ds1-group-number: -----> {0}:
line-power: -----> {disabled}:
timeslot-assignment: -----> {0}:
transmit-clock-adaptive-quality: --> {stratum3}:
.....
Save changes? [s]ave, [c]hange or [q]uit:  s
Record updated.

```

- 5 Configure the system to use the clock, as explained in [MXK clocking on page98](#).

Connect an external ring source

The MTAC card provides support for an external ring source to provide ringing voltage for the system.

Caution: When connecting the external ring source, observe the following:

If the external ring generator is an internal -48V reference (non-isolated), connect the top pin to the ringing voltage using a minimum 22 AWG wire, and leave the bottom pin unconnected. See [Figure63 on page 745](#).

If the external ring generator requires an external -48V reference (isolated), connect the top pin to the ringing voltage using a minimum 22 AWG wire and the bottom pin to -48V on the ring source. See [Figure64 on page 745](#).

Figure 63: Connecting a non-isolated ring source

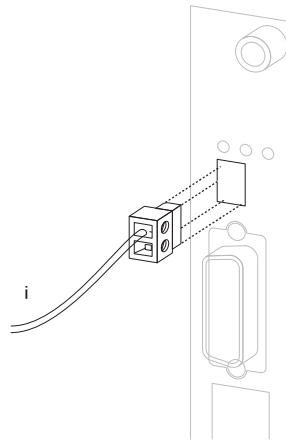
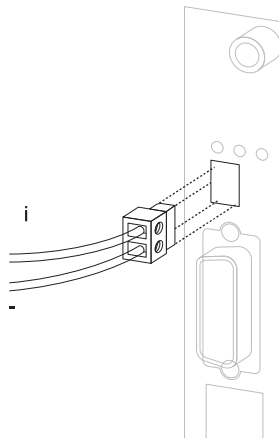


Figure 64: Connecting an isolated ring source



After connecting the ring source, update the **system** profile to specify an external ring source:

zSH> **update system 0**

system 0

Please provide the following: [q]uit.

syscontact: -----> {Zhone Global Services and Support 7001 Oakport Road Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113 support@zhone.com}:

sysname: -----> {Zhone Mxk}:

syslocation: -----> {Oakland}:

enableauthtraps: -----> {disabled}:

setserialno: -----> {0}:

zmsexists: -----> {false}:

zmsconnectionstatus: --> {inactive}:

zmsipaddress: -----> {0.0.0.0}:

configsyncexists: -----> {false}:

configsyncoverflow: --> {false}:

configsyncpriority: --> {high}:

configsyncaction: -----> {noaction}:

configsyncfilename: --> {}:

configsyncstatus: -----> {syncinitializing}:

configsyncuser: -----> {}:

configsyncpasswd: -----> {}:

numshelves: -----> {1}:

shelvesarray: -----> {}:

numcards: -----> {3}:

ipaddress: -----> {0.0.0.0}:

alternateipaddress: --> {0.0.0.0}:

countryregion: -----> {us}:

primaryclocksource: --> {0/0/0/0/0}:

ringsource: -----> {internalringsourcelabel} ~~externalringsourcelabel~~

revertiveclocksource: --> {true}

voicebandwidthcheck: --> {false}

alarm- levels- enabled: --> {critical+major+minor+warning}

userauthmode: -----> {local}

radiusauthindex: -----> {0}

secure: -----> {disabled}

webinterface: -----> {enabled}

options: -----> {NONE(0)}

.....

Save changes? [s]ave, [c]hange or [q]uit:

Record updated.

MTAC cards pinouts

This section lists the pinouts for the following interfaces on the MTAC cards:

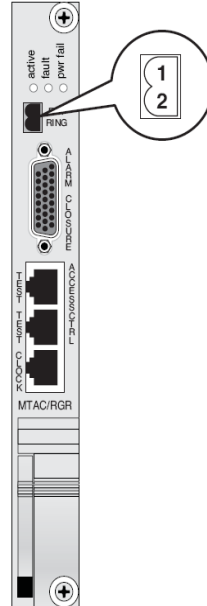
- [External ring generator input port pinouts](#)
- [External alarm sense pinouts](#)
- [Examples of alarms with specific pinouts](#)
- [Metallic test access port pinouts](#)
- [External test set control port pinouts](#)

- [External clock input port pinouts](#)

External ring generator input port pinouts

The MTAC cards provide an external ring generator input port for access to external ring generator.

Figure 65: MTAC/RING-ENH and MTAC/RING cards external ring generator input connector pinouts



[Table96](#) lists the pinouts for the external ring generator.

Table 96: External ring generator pinouts

Pin	Function
1	Ring Power Input
2	-48V Output

External alarm sense pinouts

The MTAC cards provide a 26-pin connector for access to external alarms.

The MTAC cards accept 48-volt inputs directly. All alarm inputs are metalically isolated using optocouplers. The MTAC/RING-ENH card takes 48 volts directly. Check with Zhone GSS for use of alarm sense contacts on Revision L or earlier MTAC/RING cards.

Figure 66: MTAC/RING and MTAC/RING-ENH cards external alarm connector pinouts

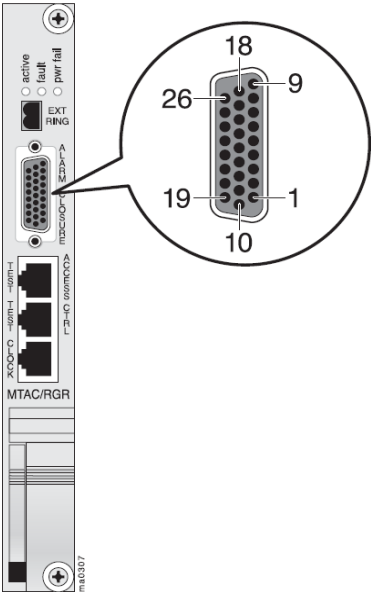


Table97 lists the pinouts for the 26-pin connector for access to external alarms.

Table 97: MTAC card external alarm connector pinouts

External alarm	Pin	Function
N/A	1	-48V supply for external contacts (fused)
1	2	Input (+)
	3	Input (-)
2	4	Input (+)
	5	Input (-)
3	6	Input (+)
	7	Input (-)
4	8	Input (+)
	9	Input (-)
5	10	Input (+))
	11	Input (-)
6	12	Input (+)
	13	Input (-)
7	14	Input (+)
	15	Input (-)

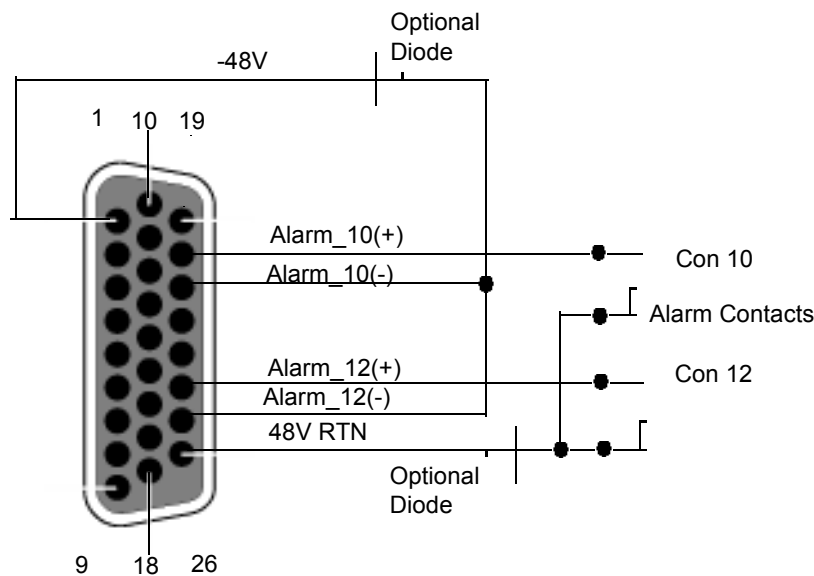
Table 97: MTAC card external alarm connector pinouts (Continued)

External alarm	Pin	Function
8	16	Input (+)
	17	Input (-)
9	18	Input (+)
	19	Input (-)
10	20	Input (+)
	21	Input (-)
11	22	Input (+)
	23	Input (-)
12	24	Input (+)
	25	Input (-)
N/A	26	48V return (+)

Examples of alarms with specific pinouts

The following example shows alarms 10 and 12 for a single MTAC/
RING-ENH (any version) or MTAC/RING (version M or greater). See [Table97](#) for other alarm pin numbers.

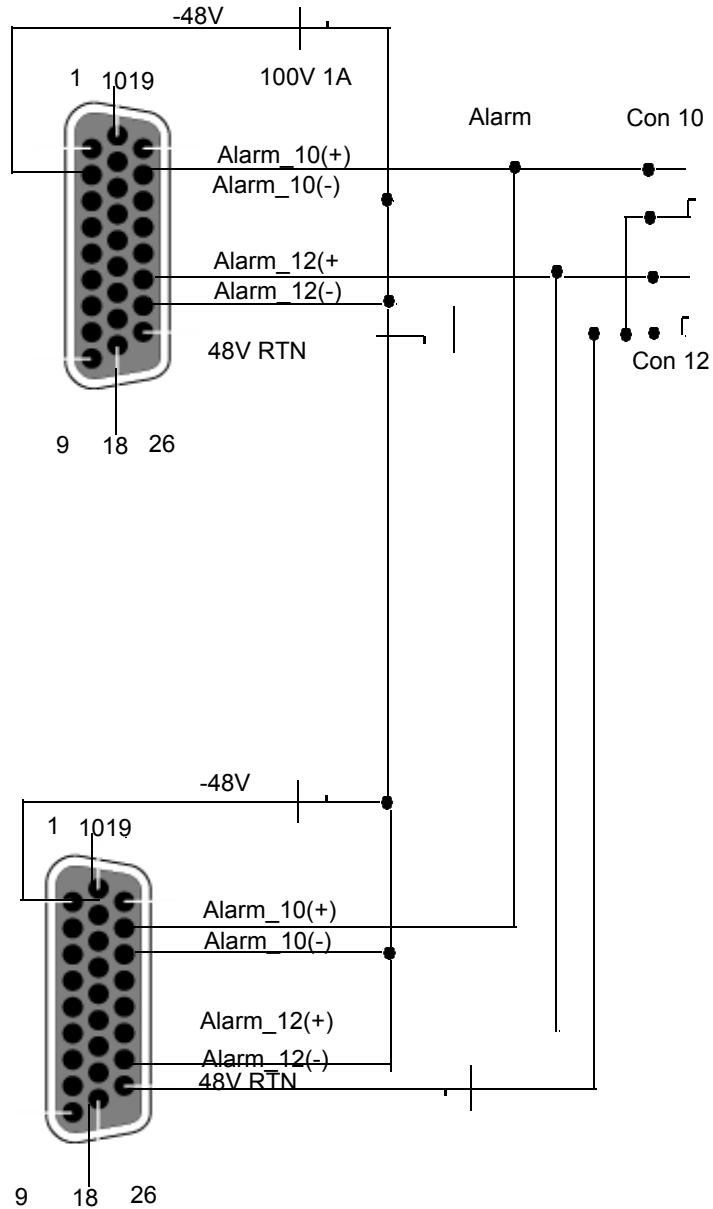
Figure 67: Single MTAC/RING-ENH or MTAC/RING Sample Connections



The following example shows alarms 10 and 12 for a redundant MTAC cards with any combination of a MTAC/RING or MTAC/RING-ENH, using board-supplied contact voltage. See [Table97](#) for other alarm pin numbers.

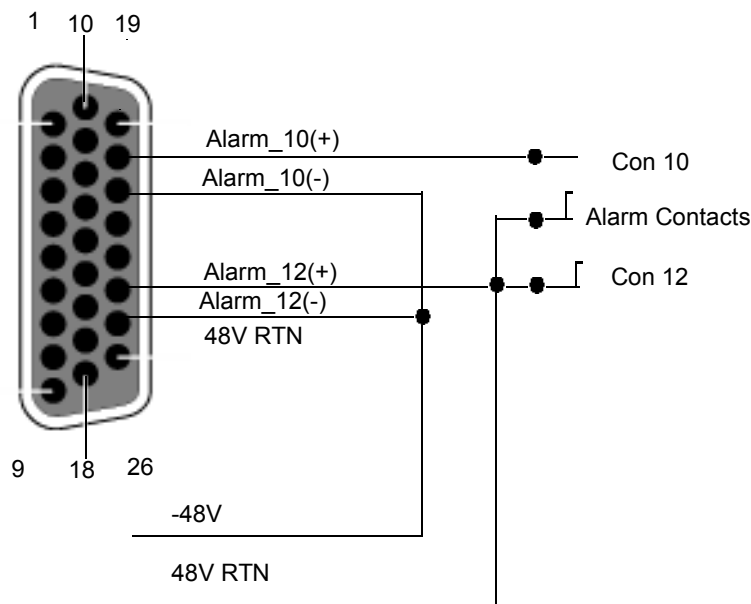
Figure 68: Redundant MTACs: Any combination of MTAC/RING, MTAC/RING-ENH Example Connections

In 4002 or Equivalent (4 Places)



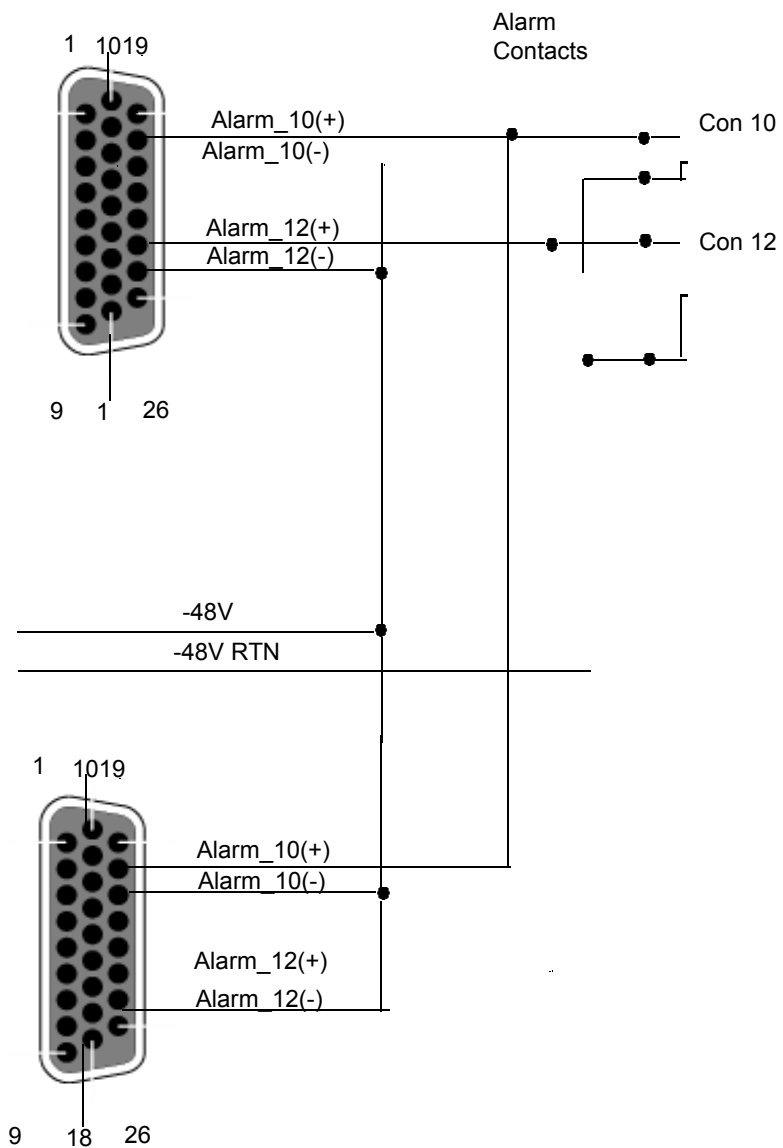
The following example shows alarms 10 and 12 for a single MTAC/RING (any version) or MTAC/RING (version M or greater). See [Table97](#) for other alarm pin numbers.

Figure 69: Single MTAC/RING-ENH or MTAC/RING with Externally Supplied Contact Voltage



The following example shows alarms 10 and 12 for redundant MTAC cards with combination of MTAC/RING-ENH (any version) or MTAC/RING (version M or greater) with an externally supplied contact voltage.

Figure 70: Redundant MTACs: Any Combination of MTAC/RING, MTAC/RING-FNH with Externally Supplied Contact Voltage



Metallic test access port pinouts

The MTAC cards provide a metallic test access port for access to an external test set.

Figure 71: MTAC/RING-ENH and MTAC/RING cards metallic test access port pinouts

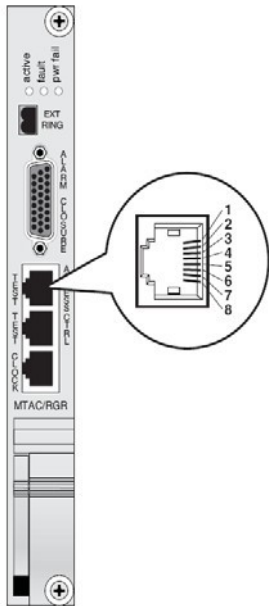


Table98 lists the pinouts for the MTAC card metallic test access port.

Table 98: MTAC card metallic test access port

Pin	Function
1	Test in tip 1
2	Test in ring 1
3	Test out tip 1
4	Test out ring 1
5	Test in tip 2
6	Test in ring 2
7	Test out tip 2
8	Test out ring 2

External test set control port pinouts

The MTAC cards provide an external test set control port to provide a control connection to the external test set.

Table99 lists the pinouts for the MTAC card external test RS232 control port.

* Factory test signals do not connect on MTAC/RING-ENH.

Table 99: MTAC card external test control port pinouts

Pin	Function
1*Reserved	
2*Reserved	
3*Reserved	
4	Signal Ground (SGND)
5	Transmitted (TxD) (Out)
6	Received (RxD)(In)
7NC	
8NC	

External clock input port pinouts

The MTAC cards provide an external clock input port to connect T1/E1 or BITS external clock reference.

[Table 100](#) lists the pinouts for the MTAC card clock port. Pinouts follow the standard RJ45 specifications with pins 1 and 2 for receive and pins 4 and 5 for transmit. Pins 6, 7, and 8 are used for 2.048 MHz square wave signals when the line-type in the DS1 profile is set to *other*.

* Connect BITS select to ground to use BITS clock input.

Table 100: MTAC card external clock pinouts

Pin	Function
1 T1/E1	Rx ring
2 T1/E1	Rx tip
3	Not used
4 T1/E1	Tx ring
5 T1/E1	Tx tip
6	BITS Select *
7BITS	clock
8	GND

15 SMALL FORM FACTOR PLUGGABLE(SFP) CONNECTORS

This chapter describes the Small Form Factor Pluggables (SFPs) and XFPs used by the MXK and covers:

- [Small form factor pluggables \(SFPs\), page757](#)
- [Insert and remove a fiber connection and an SFP, page762](#)
- [Insert and remove a dual bi-directional SFP and fiber connector, page763](#)

Small form factor pluggables (SFPs)

Zhone Technologies supports a variety of small form factor pluggables (SFPs) and XFPs which are selected depending on the protocol, fiber type and distance requirements. This section covers:

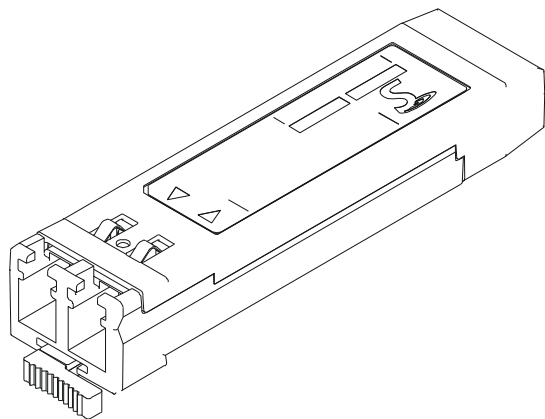
- [SFP and XFP overview, page757](#)
- [Supported SFPs, page758](#)
- [SFP specifications, page758](#)
- [XFP specifications, page760](#)
- [GPON Class B+ SFPs, page761](#)

SFP and XFP overview

SFPs (optical transceivers) and XFPs are high performance integrated duplex data links for bi-directional communication over multimode or single mode optical fiber. All Zhone Technologies SFPs are equipped with LC receptacles, which are compatible with the industry standard LC connector. These SFP transceivers measure 0.532 inches in width and provide double port densities by fitting twice the number of transceivers into the same board space as a 1x9 transceiver. All supported SFPs are hot-swappable, therefore enabling SFPs to be easily changed regardless of whether the power is on.

Furthermore, this opto-electronic transceiver module is a class 1 laser product compliant with FDA Radiation Performance Standards, 21 CFR Subchapter J. This component is also class 1 laser compliant according to International Safety Standard IEC-825-1.

Figure 72: Small form factor pluggable



Supported SFPs

The MXK supports four types of Gigabit Ethernet SFPs and one type of XFP:

- GE-SFP-SX: This is a 850 nm, multimode SFP, used in applications that are up to 5km.
- GE-SFP-LX: This is a 1310 nm, singlemode SFP, used in applications that are up to 10km.
- GE-SFP-ZX: This is a 1550 nm, singlemode SFP, used in applications that are up to 80km.
- GE-SFP-TP: This is a twisted pair SFP for access to a twisted pair GigaBit Ethernet network. It supports data rates of up to 1.25 Gbps over distances of 100 m (per IEEE 802.3).
- FE-SFP-LX: This is a 1310 nm, singlemode SFP used in applications that are up to 10km for 100 Mbps.
- MXK-10GE-XFP-LR: This is a 1310 nm, singlemode XFP, long-reach (10KM) duplex LC/UPC supporting 10 Gbps Ethernet.
- MXK-10GE-XFP-40KM-1550: XFP long-reach (40KM), single mode, 1550NM, duplex LC/UPC, supporting 10 Gbps Ethernet; I-TEMP.

The SFPs are supported to the uplink card and the dual-slot Active Ethernet card and the XFPs are supported for the uplink card.

SFP specifications

[Table101](#) describes the optical SFP specifications.

Table 101: SFP specifications

Specification	SX	LX	ZX	LX (FE)
Data rate	1.062 to 1.25 Gbps	1.062 to 1.25 Gbps	1.062 to 1.25 Gbps	100 Mbps
Fiber Interface	G.652	G.652	G.652	G.652
Operating wavelength range	830-860 nm	1274-1360 nm	1535-1565 nm	1270-1355 nm
Maximum distance supported	500 meters	10 km	80 km	10 km
Transmitter				
Source type	multimode	singlemode	singlemode	singlemode
Power -9.5dBm	(minimum) 0 dBm (maximum)	-9 dBm (minimum) -3 dBm (maximum)	2 dBm (typical)	-15 dBm (minimum) -8 dBm (maximum)
Operating temperature	min 0° C, max 70° C	min 0° C, max 70° C	min 0° C, max 70° C	min 0° C, max 70° C
Spectral characteristics: max. -20 dB width	0.85 nm	4 nm	1 nm	7.7 nm
Minimum extinction ratio	9 dB	9 dB	9 dB	9 dB
Relative intensity noise (RIN) (max.)	-117 dB	-120 dB	-120 dB	—
Optical Rise/Fall Time	300 ps	260 ps	260 ps	3 ns
Deterministic jitter (max.)	85 ps	80 ps	—	0.305 UI
Total Jitter Output (pk-pk) (max.)	251 ps	227 ps	200 ps	0.40 UI
Receiver				
Sensitivity	-17 dBm (minimum) 0 dBm (maximum)	-20 dBm (minimum) -3 dBm (maximum)	-24 dBm (minimum) 0 dBm to -3 dBm (maximum) with damage threshold at 6 dBm	-28 dBm
Reflectance at the receiving point	—	-14 dB	-14 dB	—
Deterministic jitter (max.)	113 ps	170 ps	—	0.305 UI
Total Jitter Output (pk-pk) (max.)	266 ps	266 ps	—	0.51 UI

Bi-directional SFPs

The SFP for the single slot active Ethernet card is a special design for double space gain from small factor pluggable (SFP) transceivers. These SFPs are a high performance, low cost solution for optical fiber communication, which consists of two LC receptacle bi-directional modules that meet MSA compliance.

There are four types of SFPs:

- 1310 nm Transmit, 1550 nm Receive, 1Gig speed
- 1550 nm Transmit, 1310nm Receive, 1 Gig speed
- 1310 nm Transmit, 1550 nm Receive, 100 M speed
- 1550 nm Transmit, 1310nm Receive, 100M speed



Note: The subscriber side connection to the SFP should have the opposite transmit and receive frequency. If a 1310 nm Transmit, 1550 nm Receive SFP is used on the single slot active Ethernet card, the other side should have a 1310 Receive and 1550 Transmit.

[Table102](#) provides the model names for the MXK bi-directional SFPs.

Table 102: Model names for bi-directional SFPs

Model	Description
MXK-AE-SFP-DL-BIDI-GE-1310-10	MXK AE DUAL BI-DIR SFP XMITTS 1310nm, RCV 1550 NM, supports 1Gig/100M; supports distance up to 10KM.
MXK-AE-SFP-DL-BIDI-GE-1550-10	MXK AE DUAL BI-DIR SFP XMITTS 1550nm, RCV 1310 NM, supports 1Gig/100M; supports distance up to 10KM.
MXK-AE-SFP-DL-BIDI-FE-1310-20	MXK AE DUAL BI-DIR SFP XMITTS 1310nm, RCV 1550 NM, supports 100M; supports distance up to 20KM.
MXK-AE-SFP-DL-BIDI-FE-1550-20	MXK AE DUAL BI-DIR SFP XMITTS 1550nm, RCV 1310 NM, supports 100M; supports distance up to 20KM.

XFP specifications

Zhone's FTLX1411M3 Small Form Factor 10Gb/s (XFP) transceivers are compliant with the current XFP Multi-Source Agreement (MSA) specification. They comply with SONET OC-192 SR-1, SDH STM 1-64.1, 10-Gigabit Ethernet 10GBASE-LR/LW per IEEE 802.3ae, 10G Fibre Channel 1200-SM-LL-L and the OIF Proposal for a Unified 10G Specification. Digital diagnosis functions are available through a 2-wire serial interface as specified in the XFP MSA. (The transceiver is RoHS compliant and lead free per Directive 2002/95/EC. A reference clock input is not required by the FTLX1411M3. If present, it will be ignored.)

The XFP product features:

- Supports 9.95Gb/s to 11.1Gb/s bit rates
- Hot-pluggable XFP footprint
- Maximum link length of 10km
- Uncooled 1310nm DFB laser
- Duplex LC connector
- Power dissipation <2.5 W

GPON Class B+ SFPs

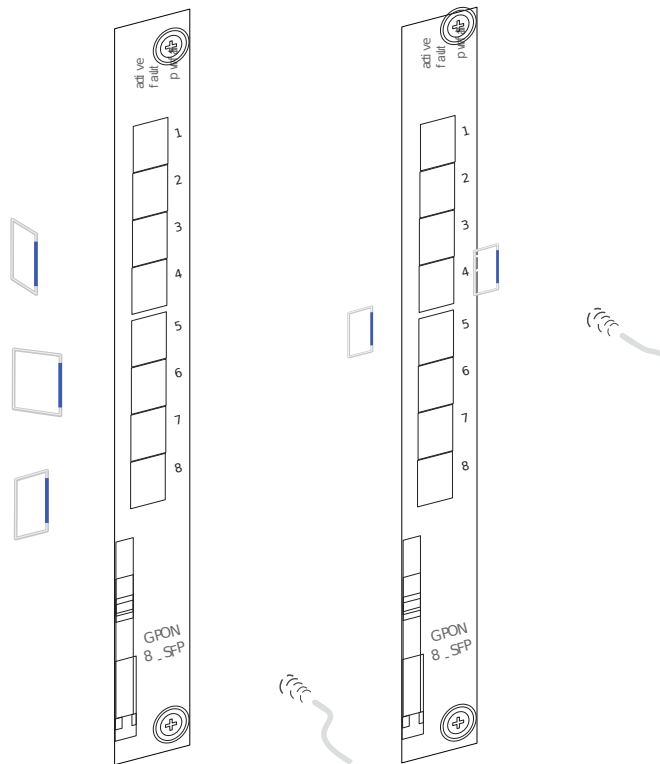
- Class B+ Optics
- 20 km reach; -28 dB link budget
- “Fast Signal Detect” feature reduces ranging overhead
- Simplified OLT “reset” timing
- 1490 nm Transmit wavelength
- 1310 nm Receive Wavelength
- 2488 Mbps downstream
- 1244 Mbps upstream Rx
- Single 3.3 V supply
- ITU-T G.984.2 compliant
- RoHS-5/6 compliant (lead exemption)
- RSSI and DDM (compliant with SFF8472 rev.9.5) supported
- operating and storage temperature -40 to +85C
- optical power 1.5W

Insert and remove a fiber connection and an SFP

Some fiber connections use Small Form Factor Pluggable (SFP) connectors. These connectors may be used with 10 Gigabit Ethernet, Gigabit Ethernet, Fast Ethernet/Gigabit Ethernet, Active Ethernet, and GPON cards.

Inserting a fiber connection and an SFP

- 1 On the SFP, push the handle wire latch in.



- 2 Slide the SFP in the port.

You should hear a slight click. Note that the SFP is not flush with the face of the card.

- 3 Insert the fiber connector into the SFP.

Removing a fiber connection and SFP

Removing SFP connectors from 10 Gigabit Ethernet, Gigabit Ethernet, Fast Ethernet/Gigabit Ethernet, Active Ethernet, or GPON cards is the reverse of installation.

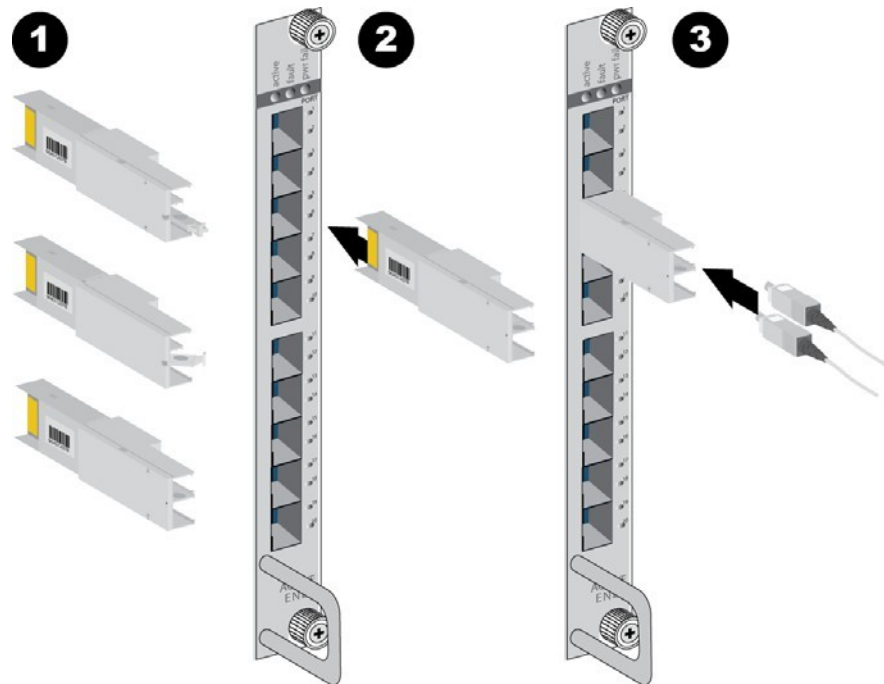
- 1 Remove the fiber connector from the SFP.
- 2 On the SFP, pull the handle wire latch outward.
- 3 Pull the SFP from the port.

Insert and remove a dual bi-directional SFP and fiber connector

Inserting a dual bi-directional SFP and fiber connector

The single slot 20 port Active Ethernet card uses dual bi-directional SFP connectors. Each physical port on the card is split into two physical ports on the SFP into which you can insert a fiber connection. Each fiber connection supports traffic in both directions.

- 1 Push in the handle latch which is between the two ports on the dual bi-directional SFP.



- 2 Slide the SFP in the port.
You should hear a slight click.



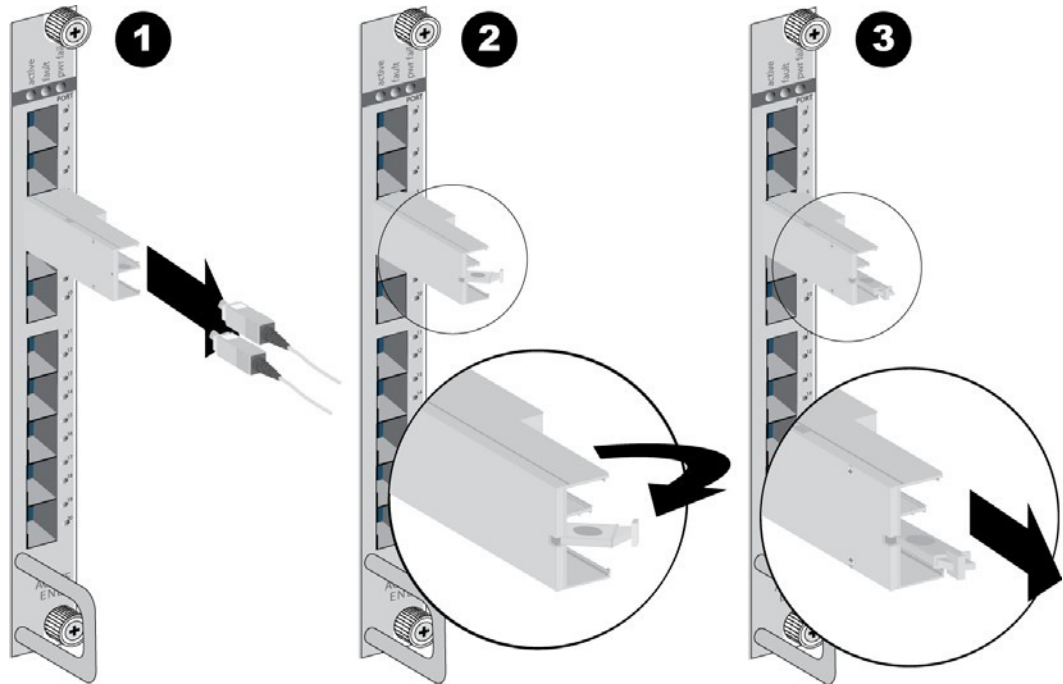
Note: The SFP is not flush with the face of the card.

- 3 Insert the fiber connector(s) into the SFP.

Removing the fiber connections and dual bi-directional SFP

Removing SFP connectors from single slot Active Ethernet cards is the reverse of installation.

- 1 Remove the fiber connector(s) from the SFP.



- 2 Pull the handle latch outward on the SFP.
- 3 Pull the SFP from the port.

INDEX

Numerics

48-port ADSL+POTS cards 390
802.1 Q-in-Q (VLAN tagging) 282
802.1p priority queues 122
802.3ah EFM OAM 696
802.3ah EFM standards 654

A

access the file system 65
activating slot cards
 slot card installation 621
Active Ethernet card, dual slot
 Ethernet interfaces 492
Active Ethernet card, dual-slot
 additional card information 491
 card type 489
 card-profile 489
 configuration 489
 overview 488
 specifications 489
Active Ethernet card, single-slot 495
 card type 497
 card-profile 497
 configuration 497
 overview 496
 specifications 497
 view additional card information 499
add users 54
admin account 55
ADSL configurable options 511
ADSL interfaces
 verifying the interface 555, 636, 691
ADSL overview 510
ADSL2+ bond card
 ADSL overview 510
 ADSL2+ bonding 566
 adsl-co-profile parameters 526
 adsl-cpe-profile parameters 531
 adsl-profile defaults 522
 adsl-profile parameters 523
 ATM data connection 569
 ATM on the ADSL2+ POTS card 569
 ATM service ranges 570
 ATM traffic descriptors 571
 Broadcom Phy-R parameters 553, 634
 cable and port pinouts 595
 card types 508
 card-profile 508
 configuration 508, 521
 configure ADSL2+ S=1/2 548
 configure Annex M 536
 configure capping train rates 543
 configure G.lite 541
 configure upstream and downstream tone
 ranges 536
 create a gbond group 566
 DSL statistics 555
 fast configuration 519
 interleaved configuration 520
 IP ToS settings and IP precedence bits 587
 MEGACO configuration 581
 MGCP configuration 579
 overview 504, 505
 POTS to VoIP configuration 572
 profiles, adsl-profile, adsl-co-profile,
 adsl-cpe-profile 521
 rate adaption 512
 Seamless Rate Adaptation 516
 SELT and DELT testing 606
 signal-to-noise (SNR) parameter 512
 SIP dial plan configuration 576
 SIP PLAR server configuration 578
 SIP server configuration 574
 SNR performance 515
 specifications 506
 transmission modes 511
 transport mode, fast or interleaved 518
 view additional card information 510, 622
 VoIP overview 572
 VoIP services 584
 VPI and VCI ranges 569
ADSL2+ bond cards
 ADSL configurable options 511
ADSL2+ bonding 566
ADSL2+ SNR 512
ADSL2+ transmission modes 511
alarms 100, 748
 external on MTAC/Ring cards 747
Annex M 536
ATM
 statistics 572
 traffic descriptor configuration rules 571
 traffic descriptors 570
ATM data connection 569
 traffic descriptors 570
ATM data connection configuration 569
ATM on the ADSL2+ POTS card 569

- ATM service ranges 570
- ATM traffic descriptors 571
- automatic baud rate adaption 666

B

- bandwidth limiting by port and service 245
- baud rate adaption 666
- bi-directional SFPs 760
- bond group bandwidth for EFM SHDSL 655
- bond group creation 658
- bond group overview for EFM 658
- bond groups, Ethernet 654
- bonded copper pairs 653
- BRAs 237, 258
- bridge show command 63
- bridge stats command 64
- bridged video 331
- bridging
 - intralinks 208
- bridging commands 310
- bridging configuration
 - bridged video 331
 - bridging configurations 274
 - downlink bridge tagged on Active Ethernet 277
 - downlink bridge untagged on Active Ethernet 276
 - downlink bridges tagged or untagged with VLAN ID 276
 - intralink bridge tagged with VLAN 0 296
 - intralink bridges staged with VLAN 0 298
 - link aggregation 304
 - Q-in-Q 282
 - Q-in-Q bridges on Active Ethernet 284
 - Q-in-Q downlink and uplink staged 287
 - tagged to staged configuration 284
 - tagged uplink bridge with VLAN ID 274
 - TLS bridges with VLAN 0 293
 - TLS bridging 300
 - uplink and downlink bridges on GPON for triple-play 279
 - uplink and downlink bridges with VLAN 0 290
 - VLAN O (VLAN wildcard) 290

C

- cable pinouts for ADSL2+ 595
- capping upstream and downstream train rates 543
- card profile

- add 51
- delete 52
- card types 51
 - Active Ethernet card, dual-slot 489
 - Active Ethernet card, single-slot 497
 - ADSL2+ bond card 508
- cards
 - Active Ethernet, dual-slot 487, 488
 - Active Ethernet, single-slot 495
 - EFM SHDSL 649
 - Ethernet uplinks 365
 - MTAC 711
 - types 650
 - viewing active redundant 713
- chassis information 56
- Class of Service (COS) 235
- clocking
 - external clock on MTAC/Ring 743
- clocking source for EFM SHDSL 652
- color aware rate limiting 252
- color blind rate limiting 248
- commands
 - rip 171
- configuration
 - RIP 171
 - verifying interfaces 555, 636, 691
- configuring IP
 - RIP 171
- configuring physical interfaces
 - verifying interfaces 555, 636, 691
- connect LP card to CPE devices 700
- Constant Bit Rate (CBR) 570
- constellation settings 668
- CoS parameters 120
- COS, in VLAN headers 235
- creates 147
- current condition minimum threshold SNR margin 672
- current condition SNR maximum threshold 672

D

- data rate range 667
- default configuration 35
- default login 35
- default password 35
- default user passwords 55
- delete admin account 55
- delete users 55

- DELT testing 606
- destination MAC swapping 243
- DHCP 116
 - configurations 126
 - DHCP relay 130
 - DHCP server options 127
 - DHCP server support 126
 - overview 126
 - server profiles 127
- DHCP configurations 126
- DHCP logging 175
- DHCP relay 130, 253
- DHCP relay agent 144, 232
- DHCP server 144
- DHCP server profiles 127
- dialing plan 576
- DNS resolver 41
- Domain Name System (DNS) 116
- downlink and uplink bridge with VLAN 0 290
- downlink bridge tagged on Active Ethernet 277
- downlink bridge untagged on Active Ethernet 276
- downlink bridges tagged or untagged with VLAN ID 276
- DSA 89
- DSL statistics on ADSL2+ bond cards 555
- dump command 67
- Dynamic Host Control Protocol (DHCP) 116
- dynamic IP filtering on a bridge
 - secure DHCP 307

E

- EFM (Ethernet in the First Mile) SHDSL 647
- EFM bond group overview 658
- EFM SHDLS card
 - specifications 649
 - switch clocking source 652
- EFM SHDSL
 - card-profile 650
- EFM SHDSL card
 - 24 NTWC card overview 27, 649
 - 24 port NTP card overview 26, 649
 - bond group bandwidth 655
 - bond groups 654
 - card configuration 650
 - card overview 649
 - create bond groups 658
 - current condition minimum threshold SNR margin 672

- current condition SNR maximum threshold 672
- deliver power and data to the CPE 700
- EFM bond group overview 658
- error monitoring 664
- Ethernet bonding 654
- Ethernet services over SHDSL links 653
- network scenario with bonded copper pairs 653
- NTP and NTWC 647
- parameters to configure SNR monitoring 673
- pinouts 699
- port statistics 695
- set SNR for target current condition or target worst case mode 675
- set SNR monitoring from CLI 676
- set wetting current 652
- SHDSL error monitoring statistics 689
- SNR crossing traps 683
- SNR maintenance mode settings 674
- SNR monitoring in the pme-profile 680
- SNR monitoring overview 671
- SNR monitoring statistics 679
- SNR pme-profile and efm-port parameters 674
- TAC testing 703
- EFM SHDSL care
 - SNR monitoring for bonded lines 671
- EFM SHDSLcard
 - connecting LP card to CPE 700
- Emergency Stand Alone (ESA) SIP support 592
- error monitoring for EFM SHDSL 664
- ESA support 592
- Ethernet interfaces 374, 492, 500
- Ethernet OAM 696
- Ethernet services over SHDSL links 653
- Ethernet uplink card 365
 - overview 366
 - pinouts 383
 - specifications 367

F

- file system 65
- fix-bit rate settings 667
- floodMulticast 302
- floodUnknown 302
- Forbid OUI 253

G

- gbond groups 566

- Generic profile
 - creation 400
 - definition 394
 - deletion 410
 - import/export 413
 - get command 61
 - GPON bridges for triple-play 279
 - GPON card
 - extended reach 428
 - Generic profile 394, 400, 410, 413
 - ME profile 394, 400, 410, 413
 - OMCI overview 392
 - Smart OMCI configuration 395
 - Smart OMCI overview 393
 - Specific profile 394, 403, 410, 413
 - GPON card overview 388
 - GPON card profiles 390
 - GPON card specifications 389
 - GPON SFPs 761
- ## H
- hookflash
 - configuring 584
 - configuring timers 584
 - host show command 63
 - host-based routing 124
 - host-based routing configurations 132
 - host-based routing for triple-play on GPON 158
 - host-based routing with an external DHCP server 144
 - host-based routing with external DHCP server 144
 - host-based routing with external DHCP server and alternate DHCP server with dhcp-relay agent 150
 - host-based routing with external DHCP server for DNS and bootp services 140
 - host-based routing with external DHCP server to provide DNS and bootp services 140
 - host-based routing with MXK as local DHCP server 137
 - host-based routing without DHCP 133
 - HTTP 87
 - HTTPS (HTTP secure) 88
- ## I
- ifindex parameter 716
 - IGMP snooping with proxy reporting 340
 - interface show command 62
 - interfaces
 - specifying type of MTAC/Ring card 711
 - intermediate agent, PPPoE 256
 - Internal ringer not detected, error message 710
 - Internet Protocol (IP) 115
 - internetworking, PPPoA-PPPoE 237, 258
 - intralink bridge tagged with VLAN 0 296
 - intralink bridges tagged with VLAN 0 298
 - Intralinks
 - configuring 208
 - IP 115, 116
 - QoS 118
 - video, configuring 315
 - IP administrative procedures 171
 - IP fallback route 169
 - IP floating interfaces 123
 - IP interface
 - Type of Service (ToS) 118
 - IP on a bridge 38
 - IP precedence bits 587
 - IP protocol
 - Routing Information Protocol (RIP) 117
 - IP protocols 116
 - Domain Name System (DNS) 116
 - Dynamic Host Control Protocol (DHCP) 116
 - IP routing 123
 - host-based routing 124
 - IP Service Level Agreement 105
 - IP services 116
 - IP statistics commands 178
 - IP ToS settings 587
 - IPSLA 105
- ## L
- LACP 349
 - available physical ports 351
 - enable Ethernet ports 355
 - line testing, MTAC 715
 - link aggregation 304, 349
 - manual link aggregation 354
 - Link Aggregation Control Protocol (LACP) 349
 - list command 57
 - log messages 36
 - log serial command 36
 - log session command 36
 - logging 71

M

- management interface
 - configure 37
 - Ethernet IP 37
 - IP on a bridge 38
 - Web user interface 43
- managing the MXK 34
- Map subscriber information to a port description
 - field 82
- mcast control list 159
- ME profile
 - creation 400
 - definition 394
 - deletion 410
 - import 410
 - import/export 413
- MEGACO configuration for ADSL2+ 581
- MGCP configuration for ADSL2+ 579
- modem train rates 667
- MTAC card
 - configuration 711
 - connectors 709
 - overview 706, 707
- MTAC/Ring card
 - configuring redundancy 711
 - external alarm contacts 748
 - ifindex 716
 - parameters 716
 - specifying line type for 711
 - test_mode 716
- MTAC/Ring external contacts 748
- multicast
 - creating control list 328, 335
- multicast control list 159
- MXK
 - default configuration 35
 - file system 65
 - file system commands 65
 - line cards overview 25
 - monitoring through the serial craft port 36
 - redundant uplinks 24
 - system administration 53
 - uplink cards overview 24
 - Zhone Management System (ZMS) 46
- MXK cards
 - MXK-ADSL2+-BCM-48A 26
 - MXK-ADSL2+-POTS-BCM-48A-2S 26
 - MXK-ADSL2+-SPLTR600-BCM-48A-2S 26

- MXK-ADSL2+-SPLTR900-BCM-48A-2S 26
- MXK-AEX20-FE/GE 25
- MXK-AEX20-FE/GE-2S 25
- MXK-EFM-SHDSL-24 NTP 26
- MXK-EFM-SHDSL-24-NTWC 27
- MXK-GPONX4-IO 25
- MXK-GPONX8-IO 25
- MXK-UPLINK-2X10GE-8X1GE 24
- MXK-UPLINK-8X1G 25
- MXK features
 - Megaco H.248 29
- MXK features 27
 - 4-port and 8-port GPON card 28
 - Active Ethernet card
 - 20-port 28
 - MGCP 28
 - SIP 29
 - uplink card redundancy 30
 - VoIP 28
- MXK overview 23

N

- network scenario with bonded copper pairs 653
- network-based routing with an external DHCP server 156
- network-based routing with the MXK as local DHCP server 154
- network-based routing without DHCP 153
- non-real-time variable bit rate (nrt-VBR) 570

O

- OAM 696
- OMCI overview 392
- Option 82 253

P

- packet-rule-record
 - bandwidth limiting by port and service 245
 - color aware rate limiting 252
 - color blind rate limiting 248
 - destination MAC swapping 243
 - DHCP relay 253
 - Forbid OUI 253
 - Option 82 253
- parameters
 - MTAC/Ring card 716

- pinouts 699
 - external alarm 747
- PME profile 665
- pme-profile (Physical Medium Entities) 665
- port access security 91
- port pinouts for ADSL2+ 595
- POTS to VoIP configuration 572
- power and data to the CPE 700
- PPP tunnel 237, 258
- PPPoA-PPPoE internetworking 237, 258
- PPPoE intermediate agent 256
- precedence bits 587

Q

- Q-in-Q 282
- Q-in-Q bridges on Active Ethernet 284
- Q-in-Q downlink and uplink tagged 287
- Q-in-Q parameters 282
- Q-in-Q tagged 284
- Q-in-Q tagged 284
- QoS 118
- QoS and traffic descriptors
- Quality of Service, see QoS

R

- RADIUS 94
- real-time variable bit rate (rt-VBR) 570
- redundancy
 - MTAC/Ring 711
 - viewing active cards 713
- regional settings 670
- reset passwords 56
- restore command 67
- RIP 117
 - configuration 171
 - configuring global defaults 171
 - rip command 171
- routed video 315
- routing 123
- Routing Information Protocol (RIP) 117
- RSA 89

S

- Seamless Rate Adaptation 516
- secure DHCP 307
- secure shell (SSH) 88

- security
 - Digital Signature Algorithm (DSA) 89
 - features 87
 - HTTPS (HTTP secure) 88
 - port access security 91
 - RSA 89
 - Secure File Transfer Protocol (SFTP) 88
 - secure shell (SSH) 88
 - SSH clients 90
 - SSH, SFTP, HTTPS 87
- SELT 606
- SELT testing 606
- serial craft interface 34
 - change port settings 34
 - initial system configuration 34
- serial craft port
 - logging sessions 36
- server-max-timer, voice-system profile 574
- setting link rates 665
- SFP 383, 492, 500
- SFP and XFP overview 757
- SFPs
 - bi-directional 760
 - GPON 761
 - insert and remove a bi-directional SFP 763
 - inserting and removing an SFP 762
 - specifications 758
 - supported SFPs 758
- SFTP 87
- SHDSL error monitoring statistics 689
- SHDSL port statistics 695
- show command 59
- Single End Loop Tests (SELT) 606
- SIP
 - connections over different networks 575
 - SIP dial plan configuration for ADSL2+ 576
 - SIP PLAR server configuration for ADSL2+ 578
 - SIP server configuration for ADSL2+ 574
 - SIP, calls not registering 574
 - sip-dialplan 576
- slot card
 - card types 51
 - provision slot cards 49
- slot cards
 - installation
 - verifying 621
- slot information 56
- slots command 50
- Small Form Factor Pluggables 383, 492, 500
- Small Form Factor Pluggables (SFPs) 757

- Smart OMCI
 - Smart OMCI overview 393
 - Smart OMCI web-interface 395
- SNMP 69
- SNR 512
- SNR crossing traps 683
- SNR current condition maximum threshold 672
- SNR current condition minimum threshold 672
- SNR for target current condition or target worst case mode 675
- SNR maintenance mode settings 674
- SNR monitoring for bonded lines 671
- SNR monitoring from CLI 676
- SNR monitoring in the pme-profile 680
- SNR monitoring overview 671
- SNR monitoring parameters 673
- SNR monitoring statistics 679
- SNR performance 515
- SNR pme-profile and efm-port parameters 674
- SNTP 68
- Specific profile
 - creation 403
 - definition 394
 - deletion 410
 - import/export 413
- SSH 87
- SSH clients 90
- tagged bridge 282
- static routes 168
- statistics
 - ATM 572
- symmetrical bridging 300
- system
 - data communications 569
- system administration
 - alarms 100
 - change default user passwords 55
 - delete admin account 55
 - delete users 55
 - DNS resolver 41
 - IP Service Level Agreement (IPSLA) 105
 - logging 71
 - reset passwords 56
 - SNMP 69
 - view chassis and slot information 56
- system configuration
 - add users 54
 - back up to a local file 68
 - back up to the network 68
 - DHCP configurations 126

- initial 34
 - restore the configuration 68
 - save and restore 67
 - user accounts 54
- system login 35
- system password 35

T

- TAC testing 703
- tagged to tagged bridge configuration 284
- TCPAM 668
- test_mode parameter 716
- TFTP server 417
- TLS bridge
 - floodUnknown and floodMulticast 302
- TLS bridge with VLAN 0 293
- ToS 118
- ToS parameters 120
- ToS settings 587
- traffic descriptors
 - configuration rules 571
 - description 570
 - QoS
- training speeds for adsl-co-profile and adsl-cpe-profile 514
- transport mode, fast or interleaved 518
- Type of Service (ToS) 118
- types, listing of cards 650, 711

U

- unspecified bit rate (UBR) 570
- update command 61
- uplink and downlink bridge with VLAN 0 290
- uplink bridge tagged with VLAN ID 274
- uplink card pinouts 383
- uplink card specifications 367
- user accounts 54

V

- video
 - configuring IP 315
 - multicast control list 328, 335
- video bridged 331
- video-source profile 158
- view additional card information 510, 622
- viewing card information 50

VLAN

Fields in the VLAN header 119

VLAN 0 290

VLAN header 119

VLAN ID and SLAN ID 282

VLAN wildcard 290, 296

voice

hookflash 584

hookflash timers 584

VoIP

hookflash, configuring 584

hookflash, configuring timers 584

SIP connections 575

VoIP overview for ADSL2+ 572

VoIP services 584

VPI and VCI ranges 569

W

Web user interface 44

wetting current 652

X

XFPs 757

specifications 760