MXK-194/198 Hardware Installation and Configuration Guide

For software version 2.2
December, 2011
Document Part Number: 830-03765-01

ZHONE™

Zhone Technologies
@Zhone Way
7195 Oakport Street
Oakland, CA 94621
USA
510.777.7000
www.zhone.com
info@zhone.com

COPYRIGHT C2000-2011 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, EZ Touch, 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

Style and notation conventions 13
Typographical conventions 14
Related documentation 14
Acronyms 15
Contacting Global Service and Support 18
Technical support 18
Hardware repair 19

Chapter 1 MXK-194/198 Overview 21
MXK-194/198 overview 21
MXK-194/198 features 24
IP and data support 24
Standards supported 24
Protocols supported 24
Management 25
MXK-194/198 chassis 26
MXK-194/198 interfaces 27
Management and other interfaces 27
FE/GE and 10GE uplink interfaces 28
Subscriber GPON interfaces 29
Small Form Factor Pluggables (XFPs and SFPs) 30
Supported SFPs and XFPs 31

Chapter 2 Preparing for installation 33
General safety precautions 33
EMC precautions 33
United States 33
Canada 34
Europe 34
Installation and servicing safety precautions 34
Power supply safety information 35
Tools you need 36
Installation precautions 37

Selecting the system location 37
Environmental specifications 38
Power requirements and specifications 39
Cabling rules 39
Power specifications 39
Grounding and isolation 40
System specifications 41

Chapter 3 System cables and connectors 43
Cabling guidelines 43
Cable descriptions 44
Chassis pinouts 45
FE/GE RJ45 port pinouts 45
Serial (craft) port pinouts 45
Alarm port pinouts 46
Connecting optical cables 47
Fiber optic maintenance and handling 48
Laser radiation 48
Handling optical fibers 49
Selecting cleaning materials 49

Chapter 4 Basic Configuration 53
Configuration overview 53
Configuration profiles 54
MXK-194/198 default configuration 54
Device settings for the MXK-194/198 55
Log into the serial (craft) port 56
Configure a management interface 57
Ethernet interface 57
Manage the MXK-19x with ZMS 60
Back up and restore the system configuration 61
Back up and restore the system configuration to the network 61
Back up and restore the system configuration locally 62
IP on a bridge for device management 63
Create a TLS IP on a bridge 64
Create an uplink IP on a bridge 66
Create a linkagg IP on a bridge 67
CPE Manager 70
Accessing the CPE’s private address, ports 70
Viewing the CPE Manager ports 75
Troubleshooting CPE Manager 77
Additional information about CPE manager 79

Web UI cut-through for GPON ONT devices 81
Map subscriber information to a port description field 83
Port description rules 83
Add, modify, list, and delete a port description 84
Add a port description to a GPON port 85
Add a port description to a bridge 85
Modify a port description 86
Port description list 87
Port description delete 87
Search port descriptions 88
MXK-194/198 security features 89
MXK-194/198 security (SSH, SFTP, and HTTPS) 89
Enable security on the MXK-194/198 89
Cipher suites 91
Tested MXK-194/198 SSH clients 91
MXK-194/198 digital signatures and public-key cryptography 92
DSA and RSA keys 92
Encryption-key commands 93
Port access security 93
Creating port-access profile entries 94
Radius support 96
Default Ethernet alarms 100
Settable alarm severity for Ethernet ports 101
Manage the MXK-194/198 with Zhone Web Graphical User Interface 103
Zhone Web GUI 103

Chapter 5 Installing the MXK-194/198 105
Unpack the MXK-194/198 system components 105
Install the chassis into a rack 106
Install mounting brackets 106
Mount the system chassis in a rack 107
Connect power to the MXK-194/198 and ground the chassis 108
Grounding requirements 108
Ground the chassis 110
Connect -48VDC Central Office power to the device 111
Verify the grounding 112
Connect alarms 113
Removable fan trays 117
MXK-194/198 system LEDs 119
Management interface LEDs 120
10GE interface LEDs 120
FE/GE interface LEDs 121
GPON interface LEDs 121
Connect optical cables 122
Fiber connections 122

Fiber management 122
Fiber guidelines and installation 122

Chapter 6 GPON on the MXK-194/198 123

GPON terminology 123
Components of GPON optical deployment networks 123
Relationship between T-conts and GEM ports 124

Bridge add, interface add, host add commands in Smart OMCI and Dynamic OMCI 126

Planning GPON networks 127

Installation testing 129

Chapter 7 IP Configuration 131

Overview 131

Terminology and concepts 133
Physical port 133
Physical interface 134
Logical interface 134
Numbered and unnumbered interfaces, floating interfaces 135
Numbered and unnumbered interfaces, floating interfaces 135

Routing types: host–based and network–based 135
Network-based (numbered) routing overview 136
Host-based (unnumbered) routing overview 137

Routing and IP addresses 139
IP addresses for downstream devices 139

IP services 141
Configuring DNS resolver 143
DHCP 145
MXK-194/198 DHCP server support 146
DHCP server profiles and scope 146
DHCP server options 147
DHCP server subnet options 148
MXK-194/198 DHCP relay 150
IP fallback route 150
RIP configuration 152
ToS, CoS, and sCoS on an IP interface 153
IP Quality of Service (QoS) 153
Fields in IP header 153
802.1p priority queues 154
Fields in the VLAN header 154
ToS, CoS, sCoS parameters 154

IP provisioning examples 156
Network-based routing 156
Static network-based routing (without DHCP) 157

Network-based routing with the MXK-194/198 as local DHCP server.....160
Network-based routing with an external DHCP server.....164
Host-based routing.....167
Static host-based routing (without DHCP) 168
Host-based routing with the MXK-194/198 as a local DHCP server 171
Static and dynamic host configuration with the same subnet 174
Host-based routing with the MXK-194/198 as a local DHCP server to provide DNS and bootp services.....175
Host-based routing with an external DHCP server 179
Host-based routing with multiple dhcp-relay agents and one DHCP server..183
Host-based routing with an external DHCP server and an alternate DHCP server with dhcp-relay agent 187
Host-based routing for data and voice services on GPON 189
IP administrative procedures 192
Modify profiles created by host/interface add commands.....192
Display hosts.....192
Filtering host show 193
Display interfaces.....194
Display routing information.....194
Displaying the routing table 194
Displaying RIP information 194
Delete hosts.....195
Delete routes 196
DHCP logging.....196
Enable DHCP logging.....196
DHCP server log messages 197
View client leases.....197
IP statistics commands.....198

Chapter 8 Bridging Configuration 203
Overview 204
Terminology and concepts 205
Physical port.....206
Physical interface.....206
Logical interface 207
Bridges and bridge interfaces.....207
VLANs and SLANs, untagged, tagged and stagged.....207
Upstream and downstream.....210
Broadcast, multicast, and unicast.....211
SLMS bridge types 213
Transparent LAN services 214
Configure a TLS bridge.....216
Asymmetric bridges.....218
Custom DHCP.....220
Custom ARP.....220
Configure an uplink bridge.....221
Bridge add and bridge-path add defaults.....222

Intralinked SLMS devices 224
Tagging operations 229
Overview 229
Common tagging operation scenarios 231
Common bridge commands 235
bridge add command 235
Verifying bridge interface settings 235
Settings for asymmetric bridges 236
Settings for symmetric bridges 237
Shaping Traffic: Class of Service Queuing 239
Configuring Class of Service 240
Filters for MXK-194/198 (packet-rule-record) 241
Overview of packet-rule-record filters 241
Create packet-rule-record filters 242
Packet rule types 243
Configure packet-rule-records 244
PPPoE with intermediate agent (bridgeinsertpppoevendortag) 245
PADI 246
PADS 246
PPPoE with intermediate agent configuration without macro defined strings 247
PPPoE with intermediate agent configuration with macro defined strings 249
Option 82 DHCP on bridge packet rule (bridgeinsertoption82) 253
Option 82 for DHCP relay overview 253
Option 82 DHCP on bridge packet rule (bridgeinsertoption82) configuration without macros 254
Option 82 DHCP on bridge packet rule (bridgeinsertoption82)configuration with macros 255
DHCP on bridge packet rules (DHCP relay, and Forbid OUI) 260
DHCP relay 260
Forbid OUI 264
Destination MAC swapping (dstmacswapdynamic and dstmacswapstatic) 265
Advanced bridging topics 267
Bridges with IGMP 267
Verifying bridge settings 269
“Denial of Service” prevention 272
Rapid Spanning Tree Protocol (RSTP) 272
RSTP port role 272
RSTP port state 274
RSTP on uplinks 274
RSTP rlinks 276
VLAN translation 281
MXK-194/198 bridging configurations 281
Configure a tagged uplink bridge with VLAN ID 282
Configure tagged or untagged downlink bridges with GEM Port IDs 283
Untagged downlink bridges on GPON interfaces 283
Configure tagged downlink bridges on GPON 284
Delete uplink and downlink bridges 285

Configure bridges using Q-in-Q (VLAN IDs and SLAN IDs) 286
Q-in-Q parameters 286
Q-in-Q bridging configurations 286
Configure TLS bridges 293
Configure TLS wire bridges 294
TLS bridge parameters floodUnknown and floodMulticast 295
floodUnknown parameter 295
floodMulticast parameter 296
Configure link aggregation bridges 298
Create a uplink bridge with link aggregation 298
Create a TLS bridge with link aggregation 299
Configure bridge loop issue prevention 299
Dynamic IP filtering on a bridge (Secure DHCP) 300
Broadcast suppression 302
Upstream multicast video support 303
Multicast VPN point-to-point service support on a wire bridge for GPON 304
Administrative commands 304
Bridge delete command 305
Bridge show/showall commands 305

Chapter 9 Video Configuration 307
MXK-194/198 bridged video 307
Configure bridged video on the MXK-194/198 308
Bridged video connection overview 308
Configure a video connection on the MXK-194/198 309
IGMP snooping with proxy reporting 313
IGMP snooping overview 314
IGMP snooping with proxy reporting 314
IGMP snooping with proxy reporting overview 314
Join and leave requests 315
IGMP snooping with proxy reporting configuration (default IP address) 316
IGMP snooping with proxy reporting with custom IP address 318
Display bridge IGMP 321
IGMPv3 messages respond for STBs 322
IGMP bridging statistics 322

Chapter 10 Ethernet Services 323
Bridging with linear 10 Gigabit Ethernet 323
Commands: port show, port up, port down, port bounce, port status 327

Chapter 11 GPON Subscriber Interfaces 329
Overview 329
Smart OMCI GPON zNID installation 329

OMCI overview 331
Smart OMCI overview....331
OMCI Profiles 331
Dynamic GEM ports 333
OMCI GPON zNID installation with Smart OMCI 334
Create a ME profile through SMART OMCI web-interface 335
Download a ME profile file to the MXK-194/198....339
Create a ME profile for the selected ONT model 340
Create Generic profiles for service plan....340
Create high speed Internet on GPON OMCI on uplink and downlink bridges343
Create an uplink and downlink bridge on GPON OMCI for video....348
Create a TLS bridge on GPON OMCI for VoIP 351
Delete the OMCI profile 353
Import and export the OMCI profile....357

Dynamic OMCI GPON zNID installation....362
Dynamic OMCI overview....363
Internal ME Profiles 364
CPE Traffic Management Profiles 365
Dynamic bridging 367
Arbitrary GEM ports 371
CPE Profiles 372
OMCI GPON zNID installation with Dynamic OMCI 376
Create High Speed Internet on Dynamic OMCI with Uplink and Downlink 376
Creating uplink and downlink bridges on Dynamic OMCI for video....382
Creating VoIP on Dynamic OMCI on uplink and downlink bridges....390
Creating PWE on Dynamic OMCI on uplink and downlink bridges....403
Creating RF on Dynamic OMCI 409
View all services on an ONU....411
Delete CPE subscriber profiles on an ONU....412
Move ONU configuration 413
VLAN translation on ONU 415
Possible bridging configuration behaviors for CPE VLAN translation 415
bridge show onu command for CPE VLAN translation....415
CPE VLAN translation on uplink and downlink bridges....416

Browser based GPON zNID installation 418
Configuring a bridge for data, GPON....419
Configuring IPTV, GPON 428
Configuring VoIP, GPON....430
Configuring Triple Play: Data, Video and Voice, GPON 435

GPON type B redundancy 436
Switchover between active and standby GPON port....441
Automatically switched....441
Manually switched from active to standby....441
Manually switched from standby to active 441
GPON redundancy configuration limitations 442

MXK-194/198 GPON using the Reg ID for provisioning 443
Configuring Reg ID 443

GPON ONU serial number format (Hexadecimal or Decimal)....445

Associate a vendor ID and a serial number with an ONU when activating the ONU 446
GPON extended reach 447
Recommendations for extended reach 447
Command to measure the distance between MXK-194/198 and ONT 448
Commands to enable extended reach 448
Received Signal Strength Indication (RSSI) and Digital Diagnostic Monitoring (DDM) 450
Manage ONU with OMCI 452
View OMCI configuration states and alarms on ONU 453
Retrieve status of subscriber facing ports 455
Administration of subscriber facing ports 455
Configurable speed of subscriber facing ports 456
Configurable Transmit Gain, Receive Gain, and Fax Mode parameters for POTS ports (Smart OMCI Only) 457
Manual upgrade on an ONU 459
Auto upgrade on an ONU 462
View the ONU upgrade status 465
Reboot an ONU 467
Re-synchronize an ONU 468
Add new services for a subscriber without affecting existing services (Smart OMCI Only) 468
Retrieve alarm information on an ONU 468
View or change trap reporting status on an ONU 468
Delete ONU configuration 469
Bandwidth allocation on T-cont 471
Configure GPON traffic profile 472
Dynamic Bandwidth Allocation (DBA) 481
GEM port creation 484
Create a GEM port 484
View the GEM port-related information 490
OMCI Statistics 491
GPON Alarms 494
GPON BIP Threshold Crossing Monitor Alarms 494
GPON High and Low Receive Power Threshold Alarms 499
Rogue ONU Alarm and RSSI Rogue ONU Alarm 502

Chapter 12 Link Aggregation Configuration 509
Link aggregation overview 509
Link aggregation and LACP 509
Link aggregation modes 509
Link resiliency 510
Configure link aggregation 510
Configure Ethernet uplink ports for manual link aggregation 511
Configure Ethernet uplink ports for LACP 512
lacp command 513

Delete a link aggregation group 513
Configure link aggregation bridges 514

Chapter 13 Diagnostics and Administration 517

IP Service Level Agreement (IPSLA) 517

MXK-194/198 logs 525
Overview 525
Default log store level 526
Improved user login notification 526
log filter command 526
Enable/disable temporary logging sessions 528
Send logging information to a syslog server 528
Create log modules 530
Log message format 532
Modify log levels 533
Use the log cache 534
Examples 534
View persistent logs 535

SNMP 535
Create SNMP community names and access lists 535
Create a community profile 536
Create community access lists 536
Configure traps 537

SNMP Statistics 537

Bridge statistics 538

Enhanced Ethernet port statistics 538

PON Statistics 555
View OLT statistics 555
View ONU statistics 564

System maintenance 566
Change the serial craft port settings 566
Rename interfaces 567
Activate or deactivate interfaces 568
Manually binding interfaces 568
Save and restore configurations 570
SNTP 571
User accounts 571
Add users 573
Change default user passwords 574
Delete users 574
Delete the admin user account 574
Reset passwords 575
SFP presence and status 575
View chassis information 578
View and set MXK-194/198 time and day 579

About This Guide

This guide is intended for use by installation technicians, system administrators, network administrators. It explains how to install the MXK-194/198 chassis and provision the system.

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.

Note: A note provides important supplemental or amplified information.

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

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

The following typographical styles are used in this guide to represent specific types of information.

Bold	Used for names of buttons, dialog boxes, icons, menus, 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 code examples for text typed by users.
Italic	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.

Related documentation

Refer to the following publication for additional information:

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

The following acronyms are related to Zhone products and may appear throughout this manual:

Table 1: Acronyms and their descriptions

Acronym	Description
802.1w	IEEE 802.1w Rapid Spanning Tree Protocol
802.3ad	IEEE 802.3ad Physical layer specifications-aggregation of multiple link segments
AC	Alternating Current
ADSL	Asymmetrical digital subscriber line
ADSL2+	Asymmetrical digital subscriber line 2+
Active E	Active Ethernet, also known as Gigabit Ethernet
APC	Angled physical contact (for fiber connector)
ARP	Address resolution protocol
ATM	Asynchronous Transfer Mode
CAT3	Category 3
CID	Channel identifier
Coax	Coaxial cable
CPE	Consumer Premises Equipment
DC	Direct Current
DDM	Digital Diagnostic Monitoring
DHCP	Dynamic Host Configuration Protocol
DSL	Digital subscriber line
EZ touch™	Zhone’s implementation for managing CPEs and zNIDs
GEM	GPON Encapsulation Method
GigE or GE	Gigabit Ethernet
GPON	Gigabit passive optical network
GTP	GPON Traffic Profile
HPNA	Home phone line networking alliance
IAD	Integrated access device
IEEE	Institute of Electrical and Electronics Engineers, Inc.
IGMP	Internet Group Management Protocol

Table 1: Acronyms and their descriptions

Acronym	Description
IMA	Inverse Multiplexing over ATM
IP	Internet Protocol
IPTV	IP (Internet Protocol) Television
LED	Light-emitting diode
LP	Line Power (+/–130 and +/–190 VDC)
MAC	Media Access Control
MALC	Multi-access line concentrator
MDU	Multiple Dwelling Unit
MIB	Management information bases
MoCA	Multimedia over Coax Alliance
MTAC	Metallic Test Access Card
MTAC-FC	Metallic Test Access Card with fan controller
OLT	Optical Line Terminator
OMCI	ONT Management and Control Interface
ONT	Optical Network Terminator
ONU	Optical Network Unit
PBX	Private branch exchange
PoE	Power over Ethernet
POTS	Plain old telephone service
PPPoE	Point-to-point protocol over Ethernet
PWE	Pseudo Wire
QoS	Quality of service
RF	Radio Frequency
RFoG	Radio Frequency over Glass
RIP	Routing Information Protocol
RSSI	Received Signal Strength Indication
SC adaptor	Subscriber connector adaptor
SHDSL	Symmetric high-bit-rate digital subscriber line
SIP	Session initiation protocol
SLMS	Single Line Multi-Service

Table 1: Acronyms and their descriptions

Acronym	Description
SNMP	Simple Network Management Protocol
SyncE	Synchronous Ethernet
T1/E1	T1 is Trunk line 1 (or DS 1, digital signal level 1). E1 is the European equivalent, though there are a number of differences between the North American T1 and the European E1.
T-Conts	Transmission Containers
TFTP	Trivial File Transfer Protocol
UPC	Ultra physical contact (for fiber connector)
UNIs	User Network Interfaces
VCI	Virtual channel identifier
VCL	Virtual channel link
VDSL2	Very high speed Digital Subscriber Lines 2
VLAN	IEEE 802.1Q Virtual LANs
VoIP	Voice over IP (Internet Protocol)
VPI	Virtual path identifier
Wi-Fi	Wireless local area network (trademark of Wi-Fi alliance)
ZMS	Zhone Management System
zNID	Zhone Network Interface Device

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) with questions about your Zhone product or other Zhone products, and 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 for a quote on a service plan. You can view service plan 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 answer 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 product(s) 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 a repair quote for anything not under warranty. Zhone requires a purchase order or credit card for out-of-warranty fees.

1 MXK-194/198 OVERVIEW

This chapter provides an overview of the MXK-194/198. It includes the following sections:

- MXK-194/198 overview, page 21
- MXK-194/198 features, page 24
- MXK-194/198 chassis, page 26
- MXK-194/198 interfaces, page 27

MXK-194/198 overview

The MXK-194/198 platform provides low-cost, high-density subscriber access concentration in the Zhone Single Line Multi-Service (SLMS) architecture.

The MXK-194/198 is a next generation design that carries data and video and voice services over GPON downlinks and Gigabit Ethernet/Fast Ethernet or, depending on the model, 10 Gigabit Ethernet uplinks. The MXK-194/198 aggregates local loop traffic from a variety of media and sends it to an upstream Fast Ethernet/Gigabit Ethernet device.

The MXK-194/198 can be deployed in Central Office environments, outdoor cabinets, or controlled environmental vaults for remote terminal applications. The MXK-194/198 is intended for restricted access locations only.

As service providers look to boost revenues and offer new multimedia services, bandwidth availability and service quality becomes crucial. The need to deliver high-definition video, faster Internet services, and on-demand content combine to require high-speed fiber-based services. Standards-based GPON technologies have become the solutions choice for delivery of full-featured Triple Play networks.

Historically, the requisite high construction and installation costs required to deploy FTTx have presented a significant barrier to its widespread adoption. Similarly, the time consuming and costly process to re-wire homes with Ethernet for IPTV have slowed deployment as well. Fiber is commonly looked to for new developments and Greenfield applications, where the costs to deploy fiber are no greater than those to deploy new copper. However, increasingly carriers are beginning the migration of FTTx into existing neighborhoods to ensure the necessary bandwidth is available to deliver a competitive suite of services. Zhone has addressed these challenges from a

service provider's perspective and developed solutions to simplify the installation of FTTx, minimize operational costs, and deliver networks fully capable of offering competitive "Triple Play" services - with no new, in-home wiring.

For existing residential communities, laying fiber all the way to the premises involves major construction as well as investment. GPON (Gigabit Passive Optical Network) is currently considered the most economic technology for FTTH (Fiber to the Home), since it supports up to 64 passive splits, optimizing the cost of the optical transceivers at the OLT, coupled with the capability to deliver 2.5 Gbps downstream and 1.25 Gbps upstream bandwidth for new services.

Despite the technical advances, which standards-based GPON offers, the initial investment is seldom justified for the delivery of just the basic POTS ("Plain Old Telephone Service") and Internet data services already provided over copper with DSL. GPON is closely coupled to triple play services, where the inclusion of IPTV or RF video service yields a new revenue stream from which the business case for the FTTH capital expenditure is justified.

Video entertainment services have evolved rapidly over the past few years from analog or digital broadcast services to enhanced high-definition (HDTV) broadcast services with new on-demand and interactive capabilities. Although new compression standards like MPEG-4 AVC lower video bandwidth per stream by one-half that of MPEG-2, increased channel offerings combined with the growing numbers of HD televisions per home are actually pushing overall bandwidth demand up. Several years ago IPTV plus internet access services typically required 15 - 23 Mbps per household, for service to two or three TVs. Today, most service providers expect to deliver HD to three to five TVs with faster Internet speeds for video downloading and gaming. Many service providers now believe the required bandwidth per home is reaching 28 - 35 Mbps, exceeding the capabilities of DSL on most copper lines. These trends in broadband service have many analysts and service providers predicting that as much as 100 Mbps will be required per home within the next 10 years.

Zhone offers a complete solution with the chassis-based MXK OLT and zNID ONT family of intelligent gateway products for every type of FTTx application. Zhone enables the intelligent, managed convergence of broadband services between the carrier and home network with up to 100 Mbps of bandwidth across a complete FTTx solution.

The MXK-194/198 includes the following models:

- MXK-194

  Four subscriber facing GPON ports, eight FE/GE uplinks.

- MXK-198

  Eight subscriber facing GPON ports, eight FE/GE uplinks.

- MXK-194-10GE

  Four subscriber facing GPON ports, two ten Gigabit Ethernet and eight FE/GE uplinks.

- MXK-198-10GE

  Eight subscriber facing GPON ports, two ten Gigabit Ethernet and eight FE/GE uplinks.

MXK-194/198 features

This section describes some key features of the MXK-194/198, including:

- IP and data support, page 24
- Standards supported, page 24
- Protocols supported, page 24
- Management

IP and data support

The MXK-194/198 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
- IP redundancy
- DHCP servers to simplify user IP address configuration and setup DHCP relay
- Numbered or unnumbered interfaces.
- Transparent bridging, VLAN bridging, intralinks, and transparent LAN service (TLS)
- Type of Service (TOS) and Class of Service (COS) processing
- IP Service Level Agreement (IPSLA)
- Secure bridging

Standards supported

The MXK-194/198 supports the following standards:

- ITU-T G.984.1- 984.4 OMCI
- IEEE 802.3 Ethernet
- IEEE 802.1p
- IEEE.802.1q
- IEEE 802.1w
- IEEE 802.3ad

Protocols supported

The MXK-194/198 supports the following protocols:

- IP host and gateway support

- RIP v 1 (RFC 1058) RIP v2 (RFC 2453)
- DHCP server (RFC 2131, 2132)
- DHCP relay with Option 82
- Bridging 802.1D support
- VLAN 802.1p/q support
- RSTP 802.1w support
- Link aggregation and LACP 802.3ad support
- Dense/sparse multicast support, IGMPv2
- Integrated access control and content protection
- RADIUS Authentication

Management

The MXK-194/198 has two primary management interfaces: a local craft RS-232 and 10/100 Base-T Ethernet.

Note: The 10/100 Base-T Ethernet interface is assigned the default IP address 192.168.10.1. This default IP address is reset if a **set2default** is performed without the restore option.

After establishing a connection to the MXK-194/198, administrators can manage the device using the Command Line Interface (CLI), Web GUI, SNMP, or ZMS.

MXK-194/198 chassis

The MXK-194/198 chassis supports:

- Eight FE/GE uplinks
- Two 10GE uplinks
- Dual redundant power inputs
- Alarm inputs
- Alarm outputs
- Port LEDs
- Removable fan tray

The MXK-194/198 cables and connectors are accessed from the rear of the chassis. Airflow through the unit is from right to left.

Figure 1 show the MXK-194/198 front faceplate with and without the ToP option. Both the MXK-194 and MXK-198 models have the same front LEDs.

Figure 2, Figure 3 show the variations of the MXK-194 models.

Figure 4, Figure 5, show the variations of the MXK-198 models.

Figure 1: MXK-194 or MXK-198 front faceplate

Figure 2: MXK-194 rear view (no T1/E1 connection)

Figure 3: MXK-194 rear view (10GE connections, no T1/E1 connection)

Figure 4: MXK-198 rear view (without ToP and T1/E1 connections)

Figure 5: MXK-198 rear view (10GE connections, no T1/E1 connection)

MXK-194/198 interfaces

This section describes the MXK-194/198 hardware, including:

- Management and other interfaces, page 27
- FE/GE and 10GE uplink interfaces, page 28
- Subscriber GPON interfaces, page 29
- Small Form Factor Pluggables (XFPs and SFPs), page 30

The interfaces are identified from the command line:

- For the FE/GE interfaces, **1-1-x-0/eth**, where x is 2 to 9.
- For the 10GE uplink interfaces, **1-1-x-0/eth**, where x is 10 or 11
- For the GPON interfaces, **1-1-x-y/gpononu**, where x is 1 to 8.

Management and other interfaces

The MXK-194/198 chassis provides the following management interfaces:

- 10/100 BaseT full duplex Ethernet port for local management or local LAN connectivity.
- RS-232 serial craft port for local management.

FE/GE and 10GE uplink interfaces

The MXK-194/198 provides Eight 100/1000 Ethernet ports with SFPs which are combo ports with RJ-45 ports, if a SFP is used for a port, the corresponding RJ-45 port cannot be used. The SFPs can be twisted pair 1000baseT or fiber (SX, LX or ZX). The 10GE models add an additional two XFP based 10GE uplinks as shown in Figure 6. For more information see *Small Form Factor Pluggables (XFPs and SFPs)* on page 30.

Figure 6: MXK-194/198 rear console ports

The optical interfaces are class 1 Laser International Safety Standard IEC 825 compliant.

Subscriber GPON interfaces

The MXK-194/198 supports four or eight ITU-T G.984 standards-based GPON ports.

GPON provides a maximum of 2.5 Gbps downstream and 1.25 Gbps upstream traffic. GPON is a point-to-multipoint architecture which may be split up to 64 subscriber ends, so the 2.5 Gbps downstream/1.25 Gbps upstream is split among the subscribers. All information is sent out to all units. Encryption keeps information private.

Figure 7: Where the MXK and the Optical Deployment Network fits in the GPON solution.

As shown in Figure 7, business logic such as downstream bandwidth is defined at the service providers access to the Internet. Upstream bandwidth characteristics are defined in the GPON traffic profiles.

The SFPs used in the MXK-194/198 GPON ports are:

- MXK-GPON-SFP-B+-RSSI
- MXK-GPON-SFP-C+-RSSI

These two SFPs support Received Signal Strength Indication (RSSI) feature. For more information about RSSI, please see Received Signal Strength Indication (RSSI) and Digital Diagnostic Monitoring (DDM), page 450.

AES encryption of 128 bits is supported on the GPON OLT chipset.

Small Form Factor Pluggables (XFPs and SFPs)

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 8: Small form factor pluggable

Supported SFPs and XFPs

The MXK-194/198 supports the following Gigabit Ethernet SFPs, GPON SFPs and 10 Gigabit Ethernet XFPs:

Table 2: Supported SFPs and XFPs

SFP/XFP Model Type	Description
Gigabit Ethernet SFPs	
Fast Ethernet/Gigabit Ethernet TP	Up to 100 meter with RJ45 connector.
Gigabit Ethernet SX	Transmits 850 nm, Receives 850 nm, up to 500 meters with duplex LC connector.
Gigabit Ethernet LX	Transmits 1310 nm, Receives 1310 nm, up to 10 kilometers with duplex LC connector.
Gigabit Ethernet EX	Transmits 1550 nm, Receives 1550 nm, up to 40 kilometers with duplex LC connector.
Gigabit Ethernet ZX	Transmits 1310 nm, Receives 1310 nm, up to 80 kilometers with duplex LC connector.
Gigabit Ethernet BX	Transmits 1310 nm, Receives 1490 nm, up to 10 kilometers with simplex LC connector.
Gigabit Ethernet BX	Transmits 1490 nm, Receives 1310 nm, up to 10 kilometers with simplex LC connector.
Gigabit Ethernet BX20	Transmits 1310 nm, Receives 1490 nm, up to 20 kilometers with simplex LC connector.
Gigabit Ethernet BX20	Transmits 1490 nm, Receives 1310 nm, up to 20 kilometers with simplex LC connector.
Gigabit Ethernet BEX	Transmits 1310 nm, Receives 1490 nm, up to 40 kilometers with simplex LC connector.
Gigabit Ethernet BEX	Transmits 1490 nm, Receives 1310 nm, up to 40 kilometers with simplex LC connector.
GPON SFPs	
GPON SFP C+	Transmits 1490 nm, Receives 1310 nm with SC/UPC connector; Up to 60 km reach; -32 dB link budget; Supports digital RSSI; ITU-T G.984.2 compliant
GPON SFP B+	Transmits 1490 nm, Receives 1310 nm with SC/UPC connector; Up to 20 km reach; -28 dB link budget; Supports digital RSSI; ITU-T G.984.2 compliant
10GE XFPs	
XFP Long Reach 40	1550 nm single mode, Duplex LC/UPC; Up to 40 kilometers
XFP Long Reach 80	1550 nm single mode, Duplex LC/UPC; Up to 80 kilometers
XFP Long Reach	1310 nm single mode, Duplex LC/UPC; Up to 10 kilometers
XFP Short Reach	850 nm multi mode, Duplex LC/UPC; Up to 300 meters

2 PREPARING FOR INSTALLATION

This chapter describes how to prepare your site for the installation of the MXK-194/198. It includes the following topics:

- General safety precautions, page 33
- Tools you need, page 36
- Installation precautions, page 37
- Selecting the system location, page 37
- Environmental specifications, page 38
- Power requirements and specifications, page 39
- System specifications, page 41

General safety precautions

This section describes the following safety precautions:

- EMC precautions, page 33
- Installation and servicing safety precautions, page 34
- Power supply safety information, page 35

The equipment is designed and manufactured in compliance with the following safety standards: UL 60950-1, EN 60950-1, IEC 60950-1, ACA TS001. However, the following additional precautions should be observed to ensure personal safety during installation or service, and to prevent damage to the equipment or equipment to which it is connected.

EMC precautions

United States

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy, and if not installed and used in accordance with the instruction manual, may cause harmful

interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which the user will be required to correct the interference at his own expense.

WARNING! The authority to operate this equipment is conditioned by the requirement that no modifications will be made to the equipment unless the changes or modifications are expressly approved by the manufacturer.

Canada

This Class A digital apparatus complies with Canadian ICES-003.

Cet appareil numérique de la classe A est conforme á la norme NMB-003 du Canada.

Europe

This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

Installation and servicing safety precautions

The precautions to take before installing or servicing the product are as follows:

Caution: Current limiting protectors

The MXK-194/198 is intended to be protected by 3-mil carbon blocks and current limiting protectors with a continuous carry current rating of 350 milliamperes. The current limiting protectors must be applied on the equipment side of the voltage limiting protector.

- Read and follow all warning notices and instructions marked on the product or included in this guide.
- Never install telephone wiring during a lightning storm.
- Never install this product in a wet location.
- Never install telephone jacks in wet locations unless the jacks are specifically designed for this purpose only.
- Never touch uninsulated telephone wires or terminals unless the telephone line has first been disconnected at the network interface.
- Use caution when installing or modifying telephone lines.
- Never attempt to service this product unless you are an authorized service technician. Doing so can expose you to dangerous high-voltage points or other risks and may result in injury or damage to the unit and void all warranties.

- The MXK-194/198 system chassis requires a dedicated ground connection to the building ground. If more than one MXK-194/198 chassis is to be installed on a rack, each one requires its own direct connection to the building ground.
- Slots and openings in the product are provided for ventilation. To ensure reliable operation of the product and to protect it from overheating, these slots and openings must not be blocked or covered.
- DO NOT allow anything to rest on the power cord and do not locate the product where anyone could step or walk on the power cord.
- Special cables, which may be required by the regulatory inspection authority for the installation site, are the responsibility of the buyer.
- When installed in the final configuration, the product must comply with the applicable Safety Standards and regulatory requirements of the country in which it is installed. If necessary, consult with the appropriate regulatory agencies and inspection authorities to ensure compliance.
- A rare phenomenon can create a voltage potential between the earth grounds of two or more buildings. If products installed in separate buildings are interconnected, the voltage potential may cause a hazardous condition. Consult a qualified electrical consultant to determine whether or not this phenomenon exists and, if necessary, implement corrective action prior to interconnecting the product.
- Install the MXK-194/198 in accordance with national and local electric codes to meet central office requirements. Consult a qualified electrical consultant.
- General purpose cables are used with this product for connection to the network. Special cables, which may be required by the regulatory inspection authority for the installation site, are the responsibility of the customer. Use a UL Listed, CSA certified (or a cable that is certified in the country in which it is being installed), minimum No. 26 AWG (.163mm2) line cord for connection to the telephone network.

Power supply safety information

Install an equipment grounding conductor not smaller in size than the ungrounded branch-circuit supply conductors as part of the circuit that supplies the product or system. Bare, covered, or insulated grounding conductors are acceptable. Individually covered or insulated equipment grounding conductors should have a continuous outer finish that is either green, or green with one or more yellow stripes. Connect the equipment-grounding conductor to ground at the service equipment.

Tools you need

The required equipment listed in Table 3 should be available before beginning the installation of the MXK-194/198 system.

Table 3: Equipment required to install the MXK-194/198 system

Qty	Equipment	Details	Use
1	Mounting shelf or rack, 19 or 23 inch width as required.	Powered as indicated in attached specifications.	MXK-194/ 198 chassis mounting
1	VT-100-compatible terminal or PC used as a VT-100 terminal emulator	Connect through craft serial port.	Commission and configuration
1	11/32-inch nutdriver	For ground stud hex nuts.	General installation
1	Pliers	N/A	General installation
1	Cable prep tools	Pressfit and crimpers	Cable installation
2	#1 and #2 Phillips-head and 1/8-inch flat-blade screwdrivers	N/A	Locking and unlocking cards, front panels and chassis brackets
2	Antistatic wrist strap	N/A	Static electricity prevention

Installation precautions

Avoid creating a hazardous condition by maintaining even weight distribution within the chassis.

Maximum operating temperature should not exceed 65^0C (149^0F). The temperature of the rack environment may be greater than ambient room temperature when the system is installed in a closed or multiunit rack assembly. Observe the maximum recommended operating temperature as indicated here.

Do not block system air vents; this will deprive the system of the airflow required for proper cooling. Sufficient clearance must exist on all sides of the rack to permit equipment access.

Zhone recommends using cabling ducts for cable routing in rack mounts.

The system ships with mounting brackets. To avoid overloading the mounting brackets, and damaging the system, do not use the MXK-194/198 chassis to support other equipment after it is mounted in the rack.

Connect the system to the power supply circuit as described in this document. Do not overload the system or power supply circuit.

Ensure that proper system grounding is performed and maintained. Use power supply connections for grounding instead of branch circuitry (such as power strips).

Selecting the system location

Ensure that the environment is free of dust and excessive moisture, not exposed to the elements or temperature extremes, and has sufficient ventilation.

Install the system in reasonable proximity to all equipment with which it will connect. Ensure that proper cable grades are used for all system and network connections. For best results, use the cables and connectors recommended in this document.

Environmental specifications

Table 4 describes the chassis environmental specifications for the MXK-194/198.

Table 4: MXK-194/198 chassis environmental specifications

Description	Specification
Chassis dimensions	4.37 cm (1.72 in.) (1U) high by 43.99 cm (17.32 in.) wide by 23.02 cm (9.06 in.) deep
Weight	8 lbs. (3.6 kg)
Operating temperature	-40^{0}C to +65^{0}C (-40^{0}F to +149^{0}F)
Storage temperature	–40^{0}C to +70^{0}C (–40^{0}F to +158^{0}F)
Operating relative humidity	5% to 85% noncondensing
Storage relative humidity	Up to 95% noncondensing
Altitude	Operating altitude: Up to -200 ft. to 16,500 ft. (-60m to 5,000m)
Airflow	Right to left

Figure 9 show the MXK-194/198 chassis dimensions.

Figure 9: MXK-194/198 chassis dimensions

Power requirements and specifications

This section includes:

- Cabling rules, page 39
- Power specifications, page 39
- Grounding and isolation, page 40

The MXK-194/198 uses a single or dual –48V DC power source.

Note: The installation site must include overcurrent protection, such as fuses or circuit breakers, that will limit current at the power input.

Cabling rules

The following are power cabling rules applicable to the MXK-194/198 system.

- Provide an appropriate disconnect device as part of the building installation for systems such as the MXK-194/198 that receive power from an external, auxiliary, or emergency source. When power is routed from a power distribution frame, the disconnect device can be used as a power cutoff (for example, an ON/OFF switch or breaker).
- Connect all disconnect devices so that they disconnect all ungrounded conductors of a DC power circuit when placed in the OFF position.
- All power cables must be rated VW-1 or higher.
- Use power cabling of 12 AWG for applications of 25 feet (7.62 m) or less from the central power distribution bus.

Power specifications

Table 5 describes the MXK-194/198 DC power specifications.

Table 5: MXK-194/198 DC power supply specifications

Descriptions	Specifications
Rated voltage	48 VDC nominal -43.75 VDC minimum -59.9 VDC maximum

Table 5: MXK-194/198 DC power supply specifications

Descriptions	Specifications
Power consumption	• MXK -194 Estimated Power, Nominal: 70W; Maximum: 88W • MXK -198 Estimated Power, Nominal: 79W; Maximum: 99W • MXK-194-10GE Estimated Power, Nominal: 84W; Maximum: 97W • MXK -198-10GE Estimated Power, Nominal: 94W; Maximum: 109W
Rated current	3A maximum
DC-input cable	AWG 12 (6 mm2) maximum
Listed circuit breaker or fuse	3 A slow blow fuse is recommended. A listed circuit breaker or fuse must be installed from a central DC power source and wired in accordance with NEC, ANSI/NFPA 70 and Canadian electrical code, Part 1, C22.1

Grounding and isolation

The MXK-194/198 uses integrated frame and logic ground system as follows:

- The MXK-194/198 system chassis and logic ground are bonded.
- The two-wire power supply feed is not connected to the chassis.
- Cable shielding is terminated on the MXK-194/198 system chassis ground.

System specifications

This section describes the following system specifications for the MXK-194/198:

- Compliance and certifications, page 41
- Management interfaces, page 41
- GPON specifications, page 42
- FE/GE uplink specifications, page 42

Table 6 describes compliance and certifications for the MXK-194/198.

Table 6: Compliance and certifications

	Specification
Safety	CSA 22.2 No. 60950-1 EN 60950-1 IEC 60950-1 UL 60950-1
EMC emissions	FCC Part 15 Class A GR-1089-Core Level 3 CE EN55022A
EMC immunity	GR-1089-Core Level 3 EN 300 386 V1.4.1
Environmental	GR-63-Core Level 3 ETS 300 019-2-x ISTA Transportation and Handling
Network	FCC Part 68 CTR-12 CTR-13 NTR-4 TSO-16

Table 7 describes the management interfaces for the MXK-194/198.

Table 7: Management interfaces

Specification	Description
Management interfaces	RS-232 serial craft port Out-of-band IP over 10/100 Base-T Ethernet

Table 8 describes the GPON specifications supported by the MXK-194/198.

Table 8: GPON specifications

Specification	Description
Density	**MXK-198** 8 port – 512 subscribers (64 subscribers per interface) **MXK-194** 4 port – 256 subscribers (64 subscribers per interface)
Physical interfaces	4-port or 8-port SFP cages capable of supporting SC-UPC fiber optic connector.
Line characteristics	Receives wavelength at a 1310nm Transmits wavelength at a 1490nm
Nominal line rate	2.5 Gbps downstream 1.25 Gbps upstream
Standard support	• ITU-T G.984

Table 9 describes the FE/GE uplink specifications for the MXK-194/198.

Table 9: FE/GE uplink specifications

Specification	Description
Density	8 ports Fast/Gigabit Ethernet
Physical interface	8 SFP cages plus 8 RJ 45 connectors
Ethernet support	100/1000/10000 Mbps Ethernet

Table 10 describes the FE/GE uplink specifications for the MXK-194/198.

Table 10: 10GE uplink specifications

Specification	Description
Density	2 ports 10 Gigabit Ethernet
Physical interface	2 XFP cages
Ethernet support	10 Gbps Ethernet

3 SYSTEM CABLES AND CONNECTORS

This chapter explains how to connect the MXK-194/198 cables and connectors. It contains the following sections:

- Cabling guidelines, page 43
- Cable descriptions, page 44
- Chassis pinouts, page 45
- Connecting optical cables, page 47
- Fiber optic maintenance and handling, page 48

Cabling guidelines

To be in compliance with NEC article 800, ensure that the power lines are placed at least two inches away from the communication cables. This can be accomplished by tie-wrapping and routing the power lines behind the rack (route the communication cables in front of the rack).

Cable descriptions

Table 11 lists specifications for the cables used with the MXK-194/198 system.

Note: Zhone recommends using 24 AWG (CAT 5) copper cables or better for 100BT Ethernet interfaces in order to achieve superior line rate performance and to improve the subscriber loop reach.

Table 11: Summary of cable specifications

Cable description	Interfaces the MXK-194/198 to	Cable type	Connector type
FE/GE uplink	FE/GE uplink device	Single Modes (SM) or MultiMode (MM) fiber or copper cable	SFP optical fiber connector or RJ45 connector
10GE uplink	10GE uplink device	Single Modes (SM) or MultiMode (MM) fiber	XFP optical fiber connector
GPON	GPON ONU/ONT devices	Single Modes (SM) fiber	SC/UPC connector
Chassis alarms output	Alarm relay contacts on the chassis	20 AWG minimum (0.8 mm) 24 AWG (0.5 mm) recommended	Blank wire in to screw terminals.
Chassis alarms input	Alarm relay contacts on the chassis	20 AWG minimum (0.8 mm) 24 AWG (0.5 mm) recommended	26 pin male D-sub connector
Management (IP)	10/100 Base-T Ethernet port	4-pair Category 5	RJ45 plug
Management (serial craft port)	8-pin RJ45	4-wire minimum 26 AWG (0.4 mm)	RJ45 plug

Chassis pinouts

This section lists the pinouts for the following interfaces on the MXK-194/198:

- FE/GE RJ45 port pinouts, page 45
- Serial (craft) port pinouts, page 45
- Alarm port pinouts, page 46

FE/GE RJ45 port pinouts

Table 12 lists the FE/GE RJ45 port pinouts.

Table 12: Ethernet port pinouts

Pin	Function
1	Tx +
2	Tx -
3	Rx +
4	Not used
5	Not used
6	Rx -
7	Not used
8	Not used

Serial (craft) port pinouts

Table 13 lists the serial (craft) port pinouts.

Table 13: Serial (craft) port pinouts

Pin	Function
1	Not used
2	Not used
3	Not used
4	Ground
5	Transmit data (with respect to the MXK-194/198) (For RS232 it is receive data with respect to the terminal (console))
6	Receive data (with respect to the MXK-194/198) (For RS232 it is transmit data with respect to the terminal (console))

Table 13: Serial (craft) port pinouts (Continued)

Pin	Function
7	Not used
8	Not used

Alarm port pinouts

Table 14: Output alarm pinouts (Pins 5, 6, & 7 of the Power ALM OUT connector)

Pin number	Description
5	NO (Normally Open)
6	Common
7	NC (Normally Closed)

Table 15: Alarm input pinouts

Pin		Pin		Pin	
1	(–) BAT -48VDC	10	5A	19	9B
2	1A	11	5B	20	10A
3	1B	12	6A	21	10B
4	2A	13	6B	22	11A
5	2B	14	7A	23	11B
6	3A	15	7B	24	12A
7	3B	16	8A	25	12B
8	4A	17	8B	26	+ 48V RTN/GND
9	4B	18	9A		

Connecting optical cables

WARNING! The single-mode fiber optic interfaces on the MXK-194/198 emit invisible laser radiation that may cause harm. When an optical cable is connected to the device, the radiation is confined to the cable and does not present a hazard. However, if you are servicing the SFPs, always use the following precautions:

- **Disconnect the fiber from the MXK-194/198 before installing or removing cables**
- **Ensure that the protective rubber tips cover the SC or LC connectors when not in use**
- **Never look directly into the optical ports**

1. Disengage the SFPs from the MXK-194/198 to ensure that the optical interface is not emitting laser radiation.
2. Remove the protective rubber tips from the SC or LC connectors on one end of the fiber optic cable or cables.
3. Remove the protective rubber tips from the SC or LC connectors on the SFPs.
4. Gently insert the SC or LC connectors on the cable into the Tx and Rx ports on the SFPs.
5. Connect the other end of the cable that is connected to the Tx connector to the Rx port on the neighboring device.
6. Connect the other end of the cable that is connected to the Rx connector into the Tx port on the neighboring device.

Fiber optic maintenance and handling

This section describes how to clean the optical connectors and receptacles used with Zhone Technologies equipment. These processes should be applied to optical components only in instances where degraded performance is evidence that the connection is contaminated.

This section contains the following information:

- Laser radiation, page 48
- Handling optical fibers, page 49
- Selecting cleaning materials, page 49

Laser radiation

Zhone equipment and associated optical test sets use laser sources that emit light energy into fiber cables. This energy is within the red (visible) and infrared (invisible) regions of the electromagnetic spectrum.

Laser products are subject to federal and state or provincial regulations, and local practices. Regulation 21 CFR 1040 of the U.S. Bureau of Radiological Health requires manufacturers to certify each laser product as Class I, II, III, or IV, depending upon the characteristics of the laser radiation emitted. In terms of health and safety, Class I products present the least hazard (none at all), while Class IV products present the greatest hazard.

Read and observe the following precautions to decrease the risk of exposure to laser radiation.

WARNING! Risk of eye damage. At all times when handling optical fibers, follow the safety procedures recommended by your company.

Although Zhone optical products have a Class I certification, hazardous exposure to laser radiation can occur when fibers connecting system components are disconnected or broken. Certain procedures carried out during testing require the handling of optical fibers without dust caps and therefore increase the risk of exposure. Exposure to either visible or invisible laser light can damage your eyes under certain conditions.

During service, maintenance, repair, or removal of cables or equipment, follow these rules:

- Avoid direct exposure to fiber ends or optical connector ends. Laser radiation may be present and can damage your eyes.
- Follow the manufacturer’s instructions when using an optical test set. Incorrect calibration or control settings can result in hazardous levels of radiation.

Handling optical fibers

When you work with optical fibers, you must take these precautions:

- Wear safety glasses when you install optical fibers.
- Clean your hands after you handle optical fibers. Small pieces of glass are not always visible and can damage your eyes. If you have a piece of a glass in your eye, get medical assistance immediately.
- Never look into an active optical fiber or a optical fiber connector opening of an active or powered-up unit.
- Prevent direct exposure to optical fiber ends or optical connector ends where you can directly access the laser signal. Do not handle pieces of optical fiber with your fingers. Use tweezers or adhesive tape to lift and discard any loose optical fiber ends.
- Wear rubber gloves when you clean optical connectors. The gloves prevent direct contact with the isopropyl alcohol and prevent contamination of the ferrules with skin oils.
- Place all optical fiber clippings in a plastic container provided for that purpose.
- Handle optical fibers with caution. Place the optical fibers in a safe location during installation.
- Fiber cannot be bent too far. Bending fiber too far will keep the optical signal from bending. You may see the light through the sheathing of the cable. These microbends may also create microfractures in the glass of the fiber resulting in signal loss.
- Protect all optical fiber connectors with clean dust caps at all times.
- Follow the manufacturer instructions when you use an optical test set. Incorrect calibration or control settings can create hazardous levels of radiation.

Fiber needs to be kept clean. Contaminants may obstruct the passing of light. Notable contaminants include

- Oil from hands
- Dust particles
- Lint
- The residue which may be left when using wet cleaning methods
- Scratches which may be from dry cleaning methods or mishandling fiber.

Selecting cleaning materials

Materials used for cleaning Zhone Technologies equipment should be high quality and suitable for the purpose.

- Disconnect the cable end to be cleaned.
- Using inert dusting gas, blow accumulated dust and debris off the cylindrical and end-face surfaces of the connector.
- Apply optical-grade isopropyl alcohol to a cleaning tissue.
- Gently wipe the tissue over the cylindrical and end face surfaces of the connector perpendicular to the cable, then fold the cloth and repeat the operation. Always use a clean tissue. Reusing the same portion of the tissue may result in recontamination.
- Dry the connector by blowing it with inert dusting gas for two seconds, holding the nozzle approximately inch from the end of the connector.
- Recap or reconnect the connector promptly to avoid contamination. Check for proper system function.

Optical cleaning kits are available from optical supply sources.

Cleaning a connector

1. Disconnect the cable end to be cleaned.
2. Using inert dusting gas, blow accumulated dust and debris off the cylindrical and end-face surfaces of the connector.
3. Apply optical-grade isopropyl alcohol to a cleaning tissue.
4. Gently wipe the tissue over the cylindrical and end face surfaces of the connector perpendicular to the cable, then fold the cloth and repeat the operation. Always use a clean tissue. Reusing the same portion of the tissue may result in recontamination.
5. Dry the connector by blowing it with inert dusting gas for two seconds, holding the nozzle approximately inch from the end of the connector.
6. Recap or reconnect the connector promptly to avoid contamination. Check for proper system function.

Cleaning a receptacle

Clean the optical ports on modules only if there is evidence of contamination or reduced performance. To minimize contamination and cleaning, keep all optical ports securely covered with a connector or a dust cap.

1. Using the extension tube supplied with the inert dusting gas, blow into the optical port to remove any accumulated dust and debris. Do not allow the tube to touch the bottom of the optical port.
2. Using a swab with a small head, such as TexWipe Microswab, and optical-grade isopropyl alcohol, wipe out the optical port.
3. Recap or reconnect the receptacle promptly to avoid contamination. Check for proper system function.

Repairing optical fibers

When an accidental break in the fiber feeder cable occurs, take the following steps:

1. Notify both central-office and field-repair personnel of the problem.
2. Identify to central-office personnel what fibers are damaged.
3. Power off all laser sources related to the damaged fibers (whether located at the central office, subscriber premises, or remote location).

4 BASIC CONFIGURATION

This chapter describes how to perform the basic configuration of the MXK-194/198. It includes the following sections:

- Configuration overview, page 53
- Device settings for the MXK-194/198, page 55
- Log into the serial (craft) port, page 56
- Configure a management interface, page 57
- IP on a bridge for device management, page 63
- Map subscriber information to a port description field, page 83
- MXK-194/198 security features, page 89
- Default Ethernet alarms, page 100
- Settable alarm severity for Ethernet ports, page 101
- Manage the MXK-194/198 with Zhone Web Graphical User Interface, page 103

Configuration overview

This section provides an overview of how to configure the MXK-194/198 and references on where to find detailed information.

1. Connect to the serial craft port. See *Log into the serial (craft) port* on page 56.
2. Configure a management interface:
   - IP interface. See *Ethernet interface* on page 57.
   - ZMS, if necessary. See *Manage the MXK-19x with ZMS* on page 60.
3. Configure uplink interfaces.
4. Configure GPON interfaces. See *GPON Subscriber Interfaces on page 329*.

Configuration profiles

The following table describes the profiles needed to perform basic MXK-194/198 configuration.

Item	Configuration profiles
GPON	**gpon-traffic-profile**: Bandwidth allocation on the upstream side of the ONU
SNMP	**community-access-profile**: SNMP access lists. **community-profile**: community-profiles. **trap-destination**: SNMP trap recipients.
System	**card-profile**: Line type settings. **ether**: Ethernet media settings. **rs232-profile**: Configures settings for serial craft ports. **system**: System contact and ZMS configuration.

MXK-194/198 default configuration

This section describes the default configuration of the MXK-194/198.

- Administrative user name is *admin*, password is *zhone*.
- A default **ip-interface-record** on *1-1-1-0-eth/ip* interface already exists. This interface is setup with the default management IP address of 192.168.10.1/24. on the 10/100 Ethernet port.
- A default **system 0** profile exists with the following configuration:
  - Authentication traps are not enabled
  - ZMS communication is not configured

Note: The 10/100 Ethernet interface is assigned the default IP address 192.168.10.1. This default IP address is reset if a **set2default** is performed without the restore option.

Device settings for the MXK-194/198

Even though the MXK-194/198 does not have cards, the device settings are stored in a profile called **card-profile**. The interface type for the MXK-194/198 always follows the convention of shelf is *1*, slot is *1* and subport is *0*. Update the **card-profile** profile to modify the device settings. Table 16 describes the device type number and software image for the MXK-194/198.

Table 16: MXK-194/198 device type

Card	Type	Name of software image
MXK-194/198	10500	*mx1u19x.bin*

Use the **slots** command to display the status of the MXK-194/198 device.

```
zSH> slots

Cards

 1: MXK 19x/MXK 198 - 8 GPON OLT, 8 FE/GE (RUNNING)
```

Enter **get card-profile** *1/1/cardtype* to view the **card-profile** parameters.

```
zSH> get card-profile 1/1/10500
card-profile  1/1/10500
sw-file-name: ----------->  {mx1u19x.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: --------->  {unknowntype}
card-atm-configuration: ->  {notapplicable}
card-line-voltage: ------>  {not-used}
maxvpi-maxvci: ---------->  {notapplicable}
card-init-string: ------->  {}
wetting-current: -------->  {disabled}
pwe-timing-mode: -------->  {none}
```

If the **card-profile** needs updating, enter the **update card-profile** *1/1/cardtype* command.

Log into the serial (craft) port

The MXK-194/198 unit provides an out-of-band RS232 D serial (craft) interface for managing the unit. To access the serial port, configure your terminal software interface with the following settings:

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

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

It is possible to manage the default MXK-194/198 by opening a Telnet or HTTP session to the autoconfig IP address of 192.168.10.1 residing on the 10/100 Ethernet management port without pre-configuring with the serial interface. See *Ethernet interface* on page 57 for how to change the default IP address.

Note: The MXK-194/198 supports 10 concurrent management sessions, 9 telnet sessions and a single local session through the serial (craft) port.

Logging in and out of the system

Log into the system (the default user name is *admin*, the default password is *zhone)*:

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

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

```
zSh> logout
```

Tip: The system automatically logs you out after a period of inactivity. The default logout time is 10 minutes, but can be changed with the **timeout** command. Refer to the *Zhone CLI Reference Guide* for information on the **timeout** command.

Enabling and disabling logging

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

```
zSh> log session off | on
```

The **log session** command only applies to the current session. You can also enable or disable logging for all serial craft port sessions using the following command:

```
zSh> log serial on | off
```

This command setting persists across system reboots.

Configure a management interface

This section describes how to configure the following interfaces to remotely manage the MXK-194/198:

- Ethernet interface, page 57
- Manage the MXK-19x with ZMS, page 60

Note: The MXK-194/198 requires the following reserved IP addresses internally:

Subnet	Host Address Range
10.254.251.0/24	10.254.251.0 to 10.254.251.255
10.254.252.0/24	10.254.252.0 to 10.254.252.255
10.254.253.0/24	10.254.253.0 to 10.254.253.255
10.254.254.0/24	10.254.254.0 to 10.254.254.255

Users must avoid assigning these IP addresses to interfaces in order to prevent potential IP addressing conflict issues.

Ethernet interface

The MXK-194/198 has a single 10/100 BaseT full duplex Ethernet interface (named **ethernet1**) designed for management traffic. By default, this interface is configured with the IP address 192.168.10.1/24 for management traffic. Connect a PC to the 10/100 Ethernet port and create a route to this interface for local device management.

Caution: The Ethernet interface must be configured before any other interfaces on the system, even if you do not intend to manage the unit over the Ethernet.

Configuring the Ethernet IP interface

To manage the MXK-194/198 through an Ethernet interface different than the default IP address for management (192.168.10.1/24), delete the default autoconfigured IP address and then add the desire interface. If the new IP address is not compatible with the address of the management PC, the connection to the device will be lost. Change the address of the management PC to be compatible with the device address.

The following example configures the IP address for the system:

```
SH> delete ip-interface-record AutoConfig/ip
ip-interface-record  AutoConfig/ip.
1 entry found.
Delete ip-interface-record AutoConfig/ip? [y]es, [n]o, [q]uit : yes
NOV 10 16:23:10: error  : 1/1/9   : tnettask: netLog(): l=158 : tNetTask:
IpIfCfgDelete: failed to get NettStateInfo if=60(I60)
ip-interface-record AutoConfig/ip deleted.

zSH> interface add 1-1-1-0/eth 192.168.254.205/24
```

Note: By default, the 10/100 Ethernet interface *1-1-1-0/eth* is assigned the IP address 192.168.10.1. This default IP address is reset if a **set2default** is performed without the restore option.

Verifying the interface

Use the **interface show** command to verify that the Ethernet interface was configured correctly:

```
zSH> interface show
1 interface
Interface      Status  Rd/Address            Media/Dest Address      IfName
-----------------------------------------------------------------------------
1/1/1/0/ip     UP      1 192.168.254.205/24  00:01:47:27:14:87       1-1-1-0-eth
-----------------------------------------------------------------------------
```

Creating a default route

The following example creates a default route using the gateway 192.168.254.254 with a cost of 1 (one):

```
zSH> route add default 192.168.254.254 1
```

Verifying the route

Use the **route show** command to verify that the routes were added:

```
zSH> route show
Destination Routing Table
Dest                    Nexthop          Cost     Owner      Fallback
-------------------------------------------------------------------------------
0.0.0.0/0               192.168.254.254  1        STATICLOW
```

```
192.168.254.0/24         1/1/1/0/ip       1        LOCAL
```

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

```
zSH> ping 192.168.254.249
PING 192.168.254.249: 64 data bytes
!!!!!
----192.168.254.249 PING Statistics----
5 packets transmitted, 5 packets received
round-trip (ms)  min/avg/max = 90/94/110
```

To stop the ping, press CTRL+C.

Manage the MXK-19x with ZMS

The **system** profile contains parameters that configure the system contact information for the MXK-194/198 and connection information for the ZMS. This profile does not need to be modified in order to manage the MXK-194/198 with ZMS.

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

CLI provisioning and ZMS

If you plan to use a script to provision the device from the CLI while it is being managed by the ZMS:

1. When the **zmsexists** parameter is set to *true*, update the **system 0** profile by changing the **zmsexists** parameter to *false* to disable partial config syncs to ZMS:

```
zSH> update system 0
system  0
syscontact: ----------->  {Zhone Global Services and Support 7001 Oakport
Street    Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113
support@zhone.com}
sysname: -------------->  {mx1u19x.bin}
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 **
numshelves: ----------->  {1}
shelvesarray: --------->  {}
numcards: ------------->  {1}
ipaddress: ------------>  {192.168.10.1}
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)}
...................
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

2 When provisioning is complete, perform a full config sync from ZMS.

Back up and restore the system configuration

You can back up and restore the system configuration on the MXK-194/198 either to or from the network or locally.

Back up the existing image and database files to the local flash card

Back up the binary file, the Web UI .tar file and the database

```
zSH> cd /card1

zSH> copy mx1u19x.bin mx1u19x.bin.bak

zSH> copy mx1u19x_http.tar mx1u19x_http.tar.bak

zSH> cd /card1/datastor

zSH> copy infotbl.v8 infotbl.v8.bak
```

Back up and restore the system configuration to the network

To back up the system configuration to the network, use the **dump** command with the following syntax:

dump network [***host filename***]

Backing up the system configuration to the network

Back up the existing system configuration to the network from the */card1* directory.

1 Create the destination directory on the TFTP server and make it writable.

2 Back up the system configuration to the TFTP server by entering **dump network** with the IP address of the server and the *filename* to create a database configuration file.

In this case, *192.168.8.21 mxk198.cfg*:

```
zSH> dump network 192.168.8.21 mxk198.cfg
```

Restoring the system configuration from the network

Restore a previously saved system configuration from a TFTP server on the network.

1. Restore the configuration that was backed up using the **file download** command.

   To restore the saved system configuration, the path name on the MXK-194/198 must be *card1/onreboot* and the file containing the system configuration information *must* be named *restore*.

```
zSH> cd /card1

zSH> file download 192.168.8.21 mxk198.cfg card1/onreboot/restore
```

2. Activate the saved system configuration with the **set2default** command.

```
zSH> set2default
```

   The **set2default** command looks in the *card1/onreboot* directory for the file *restore* and restores the database located in the *restore* file.

Back up and restore the system configuration locally

To back up the system configuration locally, create a new directory at the */card1* level, in this case *backup*, to save the system configuration file locally on the MXK-194/198.

Backing up the system configuration locally

1. Create a backup directory from the */card1* directory, in this case *backup*.

```
zSH> cd /card1

zSH> mkdir backup
```

2. Back up the system configuration to a file named *restore* with the **dump** command to create a database configuration file.

```
zSH> cd /card1

zSH> dump file backup/restore
```

   This is the file used to restore the system configuration. The *restore* file can be validated using the **restore validate** <*filename*> command.

Note: The *restore* filename is reserved for the backup and restore process and should not be used outside of this process.

Restoring the system configuration locally

Restore a previously saved system configuration from a backup file on the MXK-194/198.

1 Copy the database configuration file from the *backup* directory to */card1/onreboot/restore*.

```
zSH> cd /card1

zSH> copy backup/restore /card1/onreboot/restore
```

2 Activate the saved system database configuration file with the **set2default** command.

```
zSH> set2default
```

The **set2default** command looks in the *card1/onreboot* directory for the file *restore* and restores the database located in the *restore* file.

IP on a bridge for device management

IP on a bridge allows you to put an IP address on a bridged VLAN for managing the MXK devices. This VLAN can be used to manage multiple MXK-194/198s or other devices. One IP on a bridge can be created per MXK-194/198.

Figure 10: IP on a bridge

Create a TLS IP on a bridge

Note: The logical port interface for IP on a bridge on the MXK-194/198 *must* be *1-1-6-0/ipobridge* for correct transmission of IP packets.

Before configuring the IP on a bridge interface, create the TLS or uplink bridge. If the bridge does not exist, the following error message is displayed.

```
zSH> interface add 1-1-6-0/ipobridge vlan 200 192.168.123.21/24
Error: Couldn't determine type of IPOBRIDGE!
       Create an 'uplink' or 'tls' bridge(s) first.
```

Creating IP on a bridge for a TLS bridge

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. Create a TLS bridge on an uplink interface:

```
zSH> bridge add 1-1-4-0/eth tls vlan 200
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth/bridge
Bridge-path added successfully
```

2. Verify the bridge:

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                      St  Table Data
-------------------------------------------------------------------------------------
tls               200          1/1/4/0/eth             1-1-4-0-eth/bridge          DWN
1 Bridge Interfaces displayed
```

3. Enter **interface add** *interface/type* with the type as *ipobridge*:

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

The **interface add** command creates an *ipobridge tls* bridge with VLAN 200 with an IP address as well as a corresponding *tls* bridge on a Ethernet port for device management.

4. Verify interface:

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
-------------------------------------------------------------------------
1/1/1/0/ip     UP      1 172.24.200.64/24    00:01:47:27:14:54       1-1-1-0-eth
1/1/6/0/ip     UP      1 192.168.8.21/24     00:01:47:27:14:54       ipobridge-200
-------------------------------------------------------------------------
```

5. Verify the *tls* bridge on an Ethernet port created by the interface **add command**:

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
tls                 200        1/1/4/0/eth              1-1-4-0-eth/bridge            DWN
tls            Tagged  200     1/1/6/0/ipobridge        ipobridge-200/bridge          UP
2 Bridge Interfaces displayed
```

6 Create the default route.

See *Creating a default route on page 58*.

Create an uplink IP on a bridge

To create IP on a bridge for the uplink bridge, you must create the uplink bridge, then add the *ipobridge* interface. The **ipobridge interface** command reads the **bridge-path** profile that is created with the **bridge add** command, understands that a uplink bridge is designated, then creates a *ipobridge* downlink.

Note: If the bridge path is not added, or *uplink* is not designated when the uplink bridge is created, adding the *ipobridge* interface creates a TLS bridge.

Creating IP on a bridge for an uplink bridge

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 64.

1 Create a bridge designating uplink and a VLAN:

```
zSH> bridge add 1-1-4-0/eth uplink vlan 64 tagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-64/bridge
Bridge-path added successfully
```

2 Verify the uplink bridge:

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
tls            Tagged  105    1/1/1/0/gpononu          1-1-1-501-gponport-105/bridge  UP  D 00:00:00:0d:51:05
tls            Tagged  105    1/1/1/0/linkagg          linkagg-1-1-105/bridge         UP  D 00:00:00:ad:51:05
upl            Tagged  64     1/1/4/0/eth              1-1-4-0-eth-64/bridge          DWN S VLAN 64 default
3 Bridge Interfaces displayed
```

3 Add the *ipobridge* interface:

```
zSH> interface add 1-1-6-0/ipobridge vlan 64 192.168.8.21/24
Created ip-interface-record ipobridge-64/ip.
```

The **interface add** command creates an *ipobridge* bridge with VLAN 64 and an IP address.

4 Verify the *ipobridge* interface:

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 192.168.254.205/24  00:01:47:27:14:87       1-1-1-0-eth
1/1/6/0/ip     UP      1 192.168.8.21/24     00:01:47:27:14:87       ipobridge-64
-------------------------------------------------------------------------------
```

5 Verify the *ipobridge* created bridges:

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
tls            Tagged  105    1/1/1/0/gpononu          1-1-1-501-gponport-105/bridge  UP  D 00:00:00:0d:51:05
tls            Tagged  105    1/1/1/0/linkagg          linkagg-1-1-105/bridge         UP  D 00:00:00:ad:51:05
upl             Tagged  64     1/1/4/0/eth              1-1-4-0-eth-64/bridge          DWN S VLAN 64 default
dwn            Tagged  64     1/1/6/0/ipobridge        ipobridge-64/bridge            UP  D 00:01:47:27:14:87
4 Bridge Interfaces displayed
```

6 Create the default route.

See *Creating a default route on page 58*.

Create a linkagg IP on a bridge

IP on a bridge allows users to put an IP address on a bridged VLAN on TLS bridges. This allows VLANs to be used to manage multiple MXK-194/198s or other devices. One IP on a bridge can be created per MXK-194/198.

See *Chapter 12, Link Aggregation Configuration* for more information on link aggregation.

Create the IP on a bridge for a link aggregation group

Create an IP on a bridge interface using the IP address of 10.11.12.13/24, and with a VLAN 777. The following example shows a sample IP on a bridge creation.

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

1 Verify the *linkagg* group.

```
zSH> linkagg show
LinkAggregations:
slot unit ifName        partner: Sys            Pri     grp ID  admin   numLinks
-------------------------------------------------------------------------------
  1   1   1-1-1-0       00:00:00:00:00:00       0x0     0x0      up      2
        links           slot    port    subport         admin
        ---------------------------------------------------------------
         1-1-2-0         1       2       0               up
         1-1-3-0         1       3       0               up
```

2 Create a TLS bridge on an *linkagg* group that will manage traffic going to the group with **bridge add**, then verify the bridge created with **bridge show**:

```
zSH> bridge add 1-1-1-0/linkagg tls vlan 777 tagged
Adding bridge on 1-1-1-0/linkagg
Created bridge-interface-record linkagg-1-1-777/bridge
Bridge-path added successfully
```

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
tls            Tagged  777    1/1/1/0/linkagg           linkagg-1-1-777/bridge        UP
1 Bridge Interfaces displayed
```

Ports 2 and 3 are now reachable from the upstream, and IP 10.11.12.13/24 can reach other upstream devices on the same VLAN.

Since ports 2 and 3 on the MXK-194/198 are members of a *linkagg* group, the TLS bridge on VLAN 777 is automatically created as a *linkagg* bridge.

3 Enter **interface add** *interface/type* with the type as *ipobridge*:

```
zSH> interface add 1-1-6-0/ipobridge vlan 777 10.11.12.13/24
Created ip-interface-record ipobridge-777/ip.
```

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

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
tls            Tagged  777    1/1/1/0/linkagg           linkagg-1-1-777/bridge        UP
tls            Tagged  777    1/1/6/0/ipobridge         ipobridge-777/bridge          UP  D 00:01:47:27:14:85
2 Bridge Interfaces displayed
```

4 Enter **interface show** to verify the IP interface and then **bridge show** to verify the bridge:

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 192.168.254.207/24  00:01:47:27:14:85       1-1-1-0-eth
1/1/6/0/ip     UP      1 10.11.12.13/24      00:01:47:27:14:85       ipobridge-777
-------------------------------------------------------------------------------
```

Deleting IP on a bridge

Delete the IP on a bridge interface, and the TLS bridge on the same VLAN when necessary.

1 Delete the IP on a bridge interface:

```
zSH> interface delete 1/1/6/0/ip
Delete complete
```

The IP on a bridge interface is deleted as well as the *ipobridge*.

```
zSH> interface show
1 interface
Interface      Status  Rd/Address            Media/Dest Address      IfName
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 192.168.254.207/24  00:01:47:27:14:85       1-1-1-0-eth
-------------------------------------------------------------------------------
```

When the IP interface is deleted, the *ipobridge* is deleted as well.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                      St  Table Data
-------------------------------------------------------------------------------------
tls            Tagged  777    1/1/1/0/linkagg          linkagg-1-1-777/bridge        UP
1 Bridge Interfaces displayed
```

2 Delete the TLS *linkagg* bridge on VLAN 777.

```
zSH> bridge delete linkagg-1-1-777/bridge vlan 777
Bridge-path deleted successfully
linkagg-1-1-777/bridge delete complete
```

The IP on a bridge feature does not support SNMP.

CPE Manager

The MXK-194/198's CPE Manager provides a means for managing customer 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 zNID family of CPE and ONT products, CPE Manager can be used with any CPE or ONT device which supports receiving an IP address via DHCP 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.

Accessing the CPE's private address, ports

To access a CPE configured using CPE Manager, access the MXK-194/198 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.

To select the ports to make available the **cpe-mgr add** command has several options depending on the selection of **the compact and security parameters:**

- **compact [full | partial | none]**

  Selection of the **compact** mode defines how many ports may be accessed using the NAT-PAT binding, the more ports are accessed per device, the fewer devices that will be able to be accessed.

- **security [enabled | disabled | default]**

  Selection of the security mode defines whether those ports will use SSH, for example HTTP or HTTPS, telnet or SSH.

A list of offsets for public ports based on the compact and security mode is given in Offsets for public ports, page 72. For more information about how offsets work, see *Additional information about CPE manager on page 79*.

The defaults for compact mode is full mode (the three port mapping). For security mode, the default is default, which means to use the security settings for the MXK chassis in system 0. For additional information about security and system 0, see *Enable security on the MXK-194/198* on page 89.

Table 17: Offsets for public ports

Public port offset	Type	Name	Compact & Security Modes				
			Full		Partial		None
			Secure Enabled	Secure Disabled	Secure Enabled	Secure Disabled	N/A (all ports)
7	TCP, UDP	ECHO	+0	+0	+0	+0	+0
20	TCP	FTP - data	—	—	—	—	+1
21	TCP	FTP - control	—	—	—	—	+2
22	TCP, UDP	SSH	+1	—	+1	—	+3
23	TCP, UDP	Telnet	—	+1	—	+1	+4
80	TCP	HTTP	—	+2	—	+3	+5
81	TCP	HTTP	—	—	—	—	+6
161	TCP, UDP for partial and none UDP for full compact mode	SNMP	+2	+2	+2	+2	+7
162	UDP	SNMP traps (upstream only)	+0	+0	+3	+3	+1
443	TCP	HTTPS	+2	—	+3	—	+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 of the local network, page 73.

The IP addresses given to CPEs follow the general guidelines:

```
<Class A network>.<Slot>.<Port number: higher order byte>.<Port number: lower order byte>
```

Note that the GPON format has the port/subport encoded into the IP address which allows 12 bits for a subport and 4 bits for the port number:

```
<class A>.<slot>.<subport upper 8 bits>.<subport lower 4 bits * 16 + port>
```

Configuring the MXK-194/198 as a CPE manager for GPON

Adding CPE manager is a little different for GPON.

1. Add a public address for the CPE manager

   `cpe-mgr add public 10.55.1.237`

2 Add a GPON zNID

The following works if the GPON GEM port already exists.

`cpe-mgr add local 1-1-3-501/gponport`

If the GPON GEM port does not exist, it can be created within the **cpe-mgr add local** command by adding **gtp <gpon-traffic-profile index>**. Make sure this GPON traffic profile is created before creating the GEM port.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile  1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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

zSH> cpe-mgr add local 1-1-3-501/gponport gtp 1
GEM port 1-1-3-501/gponport created
  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: 1-1-3-501-gponport-7/ip
```

Changing the VLAN of the 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 command. Note that if you change the VLAN you would need to change the VLAN settings of all the CPEs. VLAN 7 is the default management VLAN setting of Zhone zNIDs and EtherXtend devices.

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`

Note: Zhone does not recommend changing the VLAN manually because Zhone CPE and zNID products use VLAN 7 as the default management VLAN.

Note: You can only manually set the VLAN settings when no CPE devices are currently configured on the network.

Changing the class A network used as the CPE manager local network

Once again the default settings should be acceptable. However if you need to change the default class A network the following command may be used. If you want to change network settings after CPEs are attached and configured you would have to delete them all before making the changes:

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.

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.

Viewing the CPE Manager ports

The **cpe-mgr show** command provides a mapping between the interface and the local IP address along with the various ports. For more information on available ports see Additional information about CPE manager, page 79.

```
zSH> cpe-mgr show
        CPE Manager public side interface:
                IP:      10.55.1.237
        CPE Manager local management network:
                IP:      1.0.0.1/8 (default) (active)
                VlanID:  7 (default)
        Managed CPE Interface Configuration:
 Interface             Local IP        ECHO   FTP   SSH  Telnt HTTP  SNMP  HTTPS
-------------------------------------------------------------------------------
1-1-1-501/gponport     1.1.31.84      51936  -     -   51937 51938 51938 -
1-1-3-501/gponport     1.1.31.83      51921  -     -   51922 51923 51923 -
1-1-2-701/gponport     1.1.43.211     51933  -    51934  -      -  51935 51935
1-1-3-701/gponport     1.1.43.201     51939 51940 5194251943 51944 51946 51947
1-1-4-701/gponport     1.1.43.243     51948  -     -   51949 51951 51951 -
1-1-5-701/gponport     1.1.43.223     51952  -    51953 -    51955 51954 51955
```

Compact mode full with security disabled.

```
zSH> cpe-mgr show local 1-1-3-501/gponport
Public IP address: 10.55.1.237
Public Access Port:
Protocol      Port
ECHO        51936
SNMP Traps  51936
Telnet      51937
HTTP        51938
SNMP        51938
Local IP Address: 1.1.31.83
```

Compact mode full with security enabled.

```
zSH> cpe-mgr show local 1-1-2-701/gponport
Public IP address: 10.55.1.237
Public Access Port:
Protocol      Port
ECHO        51939
SNMP Traps  51939
SSH         51940
HTTPS       51941
SNMP        51941
Local IP Address: 1.1.43.211
```

Compact mode none. Note that since all ports are available security mode is not applicable in this case.

```
zSH> cpe-mgr show local 1-1-3-701/gponport
Public IP address: 10.55.1.237
Public Access Port:
Protocol      Port
ECHO        51942
SNMP Traps  51943
FTP         51943/51944
SSH         51945
Telnet      51946
HTTP(80)    51947
HTTP(81)    51948
SNMP        51949
HTTPS       51950
Local IP Address: 1.1.43.201
```

Compact mode partial with security disabled.

```
zSH> cpe-mgr show local 1-1-4-702/gponport
Public IP address: 10.55.1.237
Public Access Port:
Protocol      Port
ECHO        51951
Telnet      51952
SNMP        51953
HTTP        51954
SNMP Traps  51954
```

```
Local IP Address: 1.1.43.243
```

Compact mode partial with security enabled.

```
zSH> cpe-mgr show local  1-1-5-701/gponport
Public IP address: 10.55.1.237
Public Access Port:
Protocol      Port
ECHO        51955
SSH          51956
SNMP        51957
HTTPS       51958
SNMP Traps  51958
Local IP Address: 1.1.43.223
```

Troubleshooting 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)

When using the defaults and the local network has not been created, the second command, **cpe-mgr add local**:

- Creates a floating **ip-interface** record with IP address of 1.0.0.1 (only created if the defaults are being used and if the record does not already exist. In other words, the first **cpe-mgr add local** if the record wasn’t created manually)
- Creates an **ip-unnumbered-record** for the floating ip-interface record (only created if the defaults are being used and if the record does not already exist. In other words, the first **cpe-mgr add local** if the record wasn’t created manually)
- Creates a **dhcp-server-subnet** for the 1.0.0.0 network (only created if the defaults are being used and if the record does not already exist. In other words, the first **cpe-mgr add local** if the record wasn’t created manually)
- 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-194/198, then Telnet from the MXK-194/198 to the device)
- Creates a **pat-bind** profile of type **cpemgr** or **cpemgrsecure**

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 (*Configuring the MXK-194/198 as a CPE manager for GPON on page 72*)contains the local IP address (`1.1.31.83`) and the CPE base port (`51921`):

```
zSH> list pat-bind
pat-bind  2
1 entry found.

zSH> get pat-bind 2
pat-bind  2
public-ipaddr: ->  {10.55.1.237}
public-port: --->  {51921}
local-ipaddr: -->  {1.1.31.83}
local-port: ---->  {3}
portType: ------>  {cpemgr}

zSH> cpe-mgr show local 1-1-3-501/gponport
        Public IP address: 10.55.1.237
        Public Access Port:
                Protocol      Port
                 ECHO        51921
                 SNMP Traps  51921
                 Telnet      51922
                 HTTP        51923
                 SNMP        51923
        Local IP Address: 1.1.31.83
```

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 interface, *1-1-3-501/gponport*, the local IP address (as shown above in the **pat-bind 2** profile) is 1.1.31.83. If you need to verify this number, do a get on the **pat-bind** profile, or use the **cpe-mgr show local** command.

Additional information about CPE manager

The first device will be accessible by the MXK-194/198'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.

Note: The examples use compact mode none. See *Configuring the MXK-194/198 as a CPE manager for GPON on page 72*. Using different variations of **compact** mode and **security** mode requires different offsets as shown in Offsets for public ports, page 72.

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 MXK-194/198'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.

CPE base port and information for added devices is shown in the cpe-mgr show display. See *Section 4, Viewing the CPE Manager ports*.

Web UI cut-through for GPON ONT devices

This section provides the configuration procedure to create hyperlinks in the MXK-194/198 Web UI that when clicked, will take you to the Web UI for device. See Figure 11 and Figure 12.

Creating a Web UI cut-through for GPON ONT devices

From the MXK-194/198 CLI:

1. Create a management interface for the MXK-194/198.
2. Create a CPE public IP using the MXK-194/198 management IP.

```
zSH> cpe-mgr add public 10.55.1.237
 CPE Manager using 10.55.1.237 for public interface.
```

3. Create a local **cpe-mgr** IP for the interface. Make sure the specified GPON traffic profile exists.

```
zSH> cpe-mgr add local 1-1-3-501/gponport gtp 1
GEM port 1-1-3-501/gponport created
  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: 1-1-3-501-gponport-7/ip
```

4. View the **pat-bind** record that was automatically created.

```
zSH> get pat-bind 2
pat-bind  2
public-ipaddr: ->  {10.55.1.237}
public-port: --->  {51921}
local-ipaddr: -->  {1.1.31.83}
local-port: ---->  {3}
portType: ------>  {cpemgr}
```

5. Verify the bond group **cpe-mgr** IP interface is UP.

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 10.55.1.237/24      00:01:47:24:08:ae  1-1-1-0-eth
1/1/3/501/ip   UP     1 [1.0.0.1]           1.1.31.83           1-1-3-501-gponport-7
-------------------------------------------------------------------------------------
```

6. From a browser, launch a Web UI to the MXK-194/198 management interface, *http://10.55.1.237*.
7. Through the WebUI, view the CPE Cut-Through URL by clicking to *Status->Service->CPE->CPE IP Hosts*.

Figure 11: The URL for the downstream device in the Web UI

ZHONE
MX-198-10GE MXK-198 10GE TOP
MX 2.2.1.226
Help
Login
Logout
Configuration
Test
Status
SYSTEM
Chassis View
Controller
Performance
Alarms
Admin Users
PORT
Ethernet
Statistics
SFP
GPON
OLT
Statistics
General Near Far
ONU
Error Statistics
OMCI
Alarms
Statistics
SFP
LinkAgg
Statistics
SERVICE
Bridging
Statistics
Address Status
CPE
IP Hosts
Member Status
Cat Binds
IP SLA
Connections
Statistics By CoS
Statistics By Interval
Routed
Statistics
DHCP Lease Status
Refresh
Note1: If both HTTP and HTTPS links of a valid CPE are disabled then the device is unreachable(no link) or initializing.
Note2: If the HTTP link is active, this indicates that the system is running in non secure mode. The CPE must be set to non secure mode in order to reach it.
Note3: If the HTTPS link is active, this indicates that the system is running in secure mode. The CPE must be set to secure mode in order to reach it.

8 Click on the HTTTP link to launch the WebUI for the GPON ONT.

Figure 12: Web UI page for the GPON zNID

Map subscriber information to a port description field

This section describes port descriptions:

- Port description rules, page 83
- Add, modify, list, and delete a port description, page 84
- Search port descriptions, page 88

Port description rules

The MXK-194/198 has a port description field, which provides a mapping between the physical port or bridge and a subscriber. This mapping improves MXK-194/198 management without requiring extra documents to provide the mapping. Port description information can be entered for ports or bridges. 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 with up to 64 characters.
- 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, page 88.

Add a port description to a GPON port

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

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

```
zSH> port description add 1-1-8-0/gponolt "Subscriber A"
```

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

To verify the port description:

```
zSH> port show 1-1-8-0/gponolt
Interface 1-1-8-0/gponolt
        Physical location:     1/1/8/0/gponolt
        Description:     Subscriber A
        Administrative status: up
```

To add a port description to a GPON ONU:

```
zSH> port description add 1-1-8-1/gpononu “business 1 555-555-5555”
```

To verify the port description enter:

```
zSH> port show 1-1-8-1/gpononu
Interface 1-1-8-1/gpononu
        Physical location:     1/1/8/1/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
     Orig
Type VLAN/SLAN VLAN/SLAN     Physical          Bridge                        St  Table Data
-------------------------------------------------------------------------------------
upl            Tagged  840   1/1/3/0/eth       1-1-3-0-eth-840/bridge        UP  S VLAN 840 default
dwn            Tagged  840   1/1/8/1/gpononu   1-1-8-501-gponport-840/bridge DWN
2 Bridge Interfaces displayed
```

Add the port description to the physical port of an existing bridge configuration, in this case the uplink bridge on Ethernet port 3:

```
zSH> port description add 1/1/3/0/eth "US Insurance Consortium, Inc."
```

Verify the port description on the uplink bridge:

```
zSH> bridge showdetail 1-1-3-0-eth-840/bridge
Bridge interface: 1-1-3-0-eth-840
        Administrative status: up         Operational status: up
        Blocked status: unblocked
        Type:upl         Tagged  840
        Data: S VLAN 840 default[U: 3600 sec, M: 150 sec, I: 70 sec]

Physical interface: 1-1-3-0/eth
        Administrative status: up         Operational status: up

Description: US Insurance Consortium, Inc.


Interface On Demand Stats State: disabled

Total Packet Statistics
        Received
                  Unicast: 0
                Multicast: 0
                Broadcast: 0
        Sent
                  Unicast: 2957638
                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

          IGMP  Received                        IGMP Transmitted
GenQuery SpecQuery v2Report    Leave   GenQuery SpecQuery v2Report    Leave
       0        0        0        0          0        0        0        0
IGMP misc: unknown=   0 errorRx=   0 actChans=   0 actHosts=   0
```

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 1-1-8-2/gpononu "Cafe Roma"
```

Verify the port description:

```
zSH> port show 1-1-8-2/gpononu
Interface 1-1-8-2/gpononu
```

```
        Description:     Cafe Roma
        Administrative status: up
```

Modify the port description:

```
zSH> port description modify 1-1-8-2/gpononu "Cafe Barrone"
```

Verify the change:

```
zSH> port show 1-1-8-2/gpononu
Interface 1-1-8-2/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 1/1/8
Interface                                Description
-------------------------------------------------------------------------------
1-1-8-0/gponolt                         subscriber A
1-1-8-0/eth                             -
1-1-8-1/gpononu                         business 1 555-555-5555
1-1-8-2/gpononu                         Cafe Barrone
1-1-8-3/gpononu                        The Carringtons
```

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 1-1-8-2/gpononu
Interface 1-1-8-2/gpononu
        Description:     Cafe Barrone
        Administrative status: up
```

To delete the port description enter:

```
zSH> port description delete 1-1-8-2/gpononu
```

To verify the deletion enter:

```
zSH> port show 1-1-8-2/gpononu
Interface 1-1-8-2/gpononu

        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 Carringtons
Results for Carringtons
Description:     Carringtons
Interface:       1-1-8-3/gponONU
Number of arguments for this command is 1

zSH> port description find "US Insurance Consortium, Inc."
Results for US Insurance Consortium, Inc.
Description:     US Insurance Consortium, Inc.
Interface:       1-1-3-0/eth
Number of arguments for this command is 1
```

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

MXK-194/198 security features

This section describes the MXK-194/198's security features including Radius support, Secure Shell (SSH), Secure File Transfer Protocol (SFTP), HTTPS, and port access security:

- MXK-194/198 security (SSH, SFTP, and HTTPS), page 89
- MXK-194/198 digital signatures and public-key cryptography, page 92
- Port access security, page 93
- Radius support, page 96

MXK-194/198 security (SSH, SFTP, and HTTPS)

This section covers security on the MXK-194/198:

- Enable security on the MXK-194/198, page 89
- Cipher suites, page 91
- Tested MXK-194/198 SSH clients, page 91

Enable security on the MXK-194/198

The **system 0** profile provides a **secure** parameter which allows only secure communication for management activities. When security is enabled, the MXK-194/198 uses the following protocols:

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

Table 18 describes which protocols are allowed when the **secure** parameter is *enabled* and which protocols are allowed when the **secure** parameter is *disabled*.

Table 18: Protocols for the secure parameter

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

Enabling security on the MXK-194/198

1 To enable the security parameter enter **update system 0** on the MXK-194/198, 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: -------------->  {mx198-10GE-TOP}:
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: ------------->  {1}:
ipaddress: ------------>  {192.168.10.1}:
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}: enabled
webinterface: --------->  {enabled}:
options: -------------->  {NONE(0)}:
....................
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Cipher suites

The MXK-194/198 supports several ciphers for SSH.

Table 19: MXK-194/198 ciphers

Cipher	Key size
aes256-cbc	32 bytes (256 bits)
rijndael256-cbc	32 bytes (256 bits)
aes192-cbc	24 bytes (192 bits)
rijndael192-cbc	24 bytes (192 bits)
aes128-cbc	16 bytes (128 bits)
rinjdael128-cbc	16 bytes (128 bits)
blowfish-cbc	16 bytes (128 bits)
3des-cbc	24 bytes (192 bits)
arcfour	16 bytes (128 bits)

Tested MXK-194/198 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-194/198 using the SSH client of your choice to encrypt the session. The MXK-194/198 supports the following:

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

MXK-194/198 digital signatures and public-key cryptography

This section describes the MXK-194/198's security features for digital signatures and public-key cryptography.

- DSA and RSA keys, page 92
- Encryption-key commands, page 93

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

DSA and RSA keys

The MXK-194/198 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.
zSH>
```

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
```

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-194/198 provides security on the UDP/TCP ports which the MXK-194/198 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 ports used for management are:

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

If you choose to restrict access to the SNMP port then there must be a rule to allow the MXK-194/198 its own SNMP access. See *Creating a port-access entry for the MXK-194/198 to maintain SNMP access on page 96*.

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 setting secure mode for SFTP and SSH in system 0. If secure is enabled which provides SSH and SFTP while limiting telnet access, but 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.

Creating port-access profile entries

To 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.

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

Creating a port-access entry for a specific IP address

To create 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.

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-194/198.

Note: To create port access protection for both HTTP and HTTPS you will need to 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.
```

Creating a port-access entry for a subnet

To create 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.

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-194/198.

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.
```

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.
```

Modifying port-access profile entries

To modify a configured **port-access** profile entry use the **update** command. The following 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.
```

Creating a port-access entry for the MXK-194/198 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-194/198 supports local and RADIUS (Remote Authentication Dial In User Service) access authentication. The MXK-194/198 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.

Table 20 shows the mapping of service-type to MXK-194/198 permissions.

Table 20: Service type mapping to MXK-194/198 permissions

Service-Type Attribute	MXK-194/198 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-194/198 with RADIUS authentication, the MXK-194/198 passes RADIUS information securely to the RADIUS server. The RADIUS server then authenticates the user and

either allows or denies access to the MXK-194/198. If access is denied and the local authentication option is also configured, the MXK-194/198 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-194/198 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 13 illustrates MXK-194/198 Radius authentication.

Figure 13: MXK-194/198 RADIUS authentication

Note: Follow the RADIUS server guidelines for RADIUS configuration instructions. For example, when using the MXK-194/198 with the FreeRadius server:

- Create only one entry in the *clients.conf* file for each subnet or individual MXK-194/198. For individual MXK-194/198s, the IP in this file must match the IP address of the outbound interface used by the MXK-194/198 to connect to the RADIUS server.
- The MXK-194/198 uses the value stored in the RADIUS *system.sysname* file for the NAS-Identifier attribute.
- The shared-secret in the MXK-194/198 radius-client profile, must exactly match the shared-secret in the RADIUS client entry.

Configuring RADIUS support

The MXK-194/198 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-194/198 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-194/198 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-194/198 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-194/198. 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 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.
```

3. Create another radius-client profile on the MXK-194/198 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: ---->  {}: 172.24.36.248
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.
```

Create additional radius-client profiles for each additional RADIUS server to be assigned to this MXK-194/198.

4 In the system profile on the MXK-194/198, set the desired user authentication method and specify the index of the radius profile to use. This examples specifies the **radiusauthindex** of 1. This index is configured with two radius-client profiles (1/1, 1/2). The MXK-194/198 first attempts authentication using the server specified in radius-client 1/1. If this authentication fails, the MXK-194/198 attempts authentication using radius-client 1/2 server. If this authentication also fails, the MXK-194/198 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-194/198 may become inaccessible if IP connectivity to the RADIUS server is lost or other changes prevent the MXK-194/198 from receiving RADIUS authentication.

After completing the RADIUS configuration, the MXK-194/198 displays console messages for RADIUS login and logout activity.

```
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: -------------->  {mx198-10GE-TOP}:
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: ------------->  {1}:
ipaddress: ------------>  {192.168.10.1}:
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.
```

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-194/198 using the default Zhone MXK-194/198 system name will appear as basicuser@mxk194 or basicuser@mxk198. The system name is configured using the sysname parameter in the **system 0** profile.

Default Ethernet alarms

Enter the **port show alarm** *interface/type* command to view the level of alarm severity on Ethernet ports. The default for Ethernet uplink ports is *critical*.

```
zSH> port show alarm 1-1-2-0/eth
-----------------------------------------------
        Interface              Alarm severity
-----------------------------------------------
        1-1-2-0/eth            CRITICAL
-----------------------------------------------
```

Settable alarm severity for Ethernet ports

The alarm severity for Ethernet ports can be set to the following levels: *critical*, *major*, *minor*, or *warning*.

Changing the alarm severity level for one Ethernet port

Use the **port config alarm** *interfaceName/type* **severity** <*severity level*> command to set the severity level on an Ethernet port.

1 View the current alarm setting on an Ethernet port.

```
zSH> port show alarm 1-1-4-0/eth
-----------------------------------------------
        Interface              Alarm severity
-----------------------------------------------
        1-1-4-0/eth            CRITICAL
-----------------------------------------------
```

2 Configure a different alarm setting on an Ethernet port.

```
zSH> port config alarm 1-1-4-0/eth severity major
Alarm severity level set for 1-1-4-0/eth is major
```

3 Verify the new alarm setting.

```
zSH> port show alarm 1-1-4-0/eth
------------------------------------------------
        Interface               Alarm severity
------------------------------------------------
        1-1-4-0/eth             MAJOR
------------------------------------------------
```

Changing the alarm severity level for multiple Ethernet ports

Use the **port config alarm** *interfaceName/type* **severity** <*severity level*> command to set the severity level on multiple Ethernet ports.

1 View the alarm levels for all Ethernet uplink ports .

```
zSH> port show alarm 1-1-*-*/eth
-----------------------------------------------
        Interface              Alarm severity
-----------------------------------------------
        1-1-9-0/eth            MAJOR
        1-1-8-0/eth            CRITICAL
        1-1-7-0/eth            CRITICAL
        1-1-6-0/eth            CRITICAL
        1-1-5-0/eth            CRITICAL
        1-1-4-0/eth            MAJOR
        1-1-3-0/eth            CRITICAL
        1-1-2-0/eth            CRITICAL
        1-1-1-0/eth            CRITICAL
-----------------------------------------------
```

2 Change the alarm setting of all Ethernet ports on the line card.

```
zSH> port config alarm 1-1-*-*/eth severity critical
Alarm severity level set for 1-1-8-0/eth is critical
Alarm severity level set for 1-1-7-0/eth is critical
Alarm severity level set for 1-1-6-0/eth is critical
Alarm severity level set for 1-1-5-0/eth is critical
Alarm severity level set for 1-1-4-0/eth is critical
Alarm severity level set for 1-1-3-0/eth is critical
Alarm severity level set for 1-1-2-0/eth is critical
Alarm severity level set for 1-1-1-0/eth is critical
```

3 Verify the alarm severity level.

```
zSH> port show alarm 1-1-*-*/eth
-----------------------------------------------
        Interface              Alarm severity
-----------------------------------------------
        1-1-9-0/eth            CRITICAL
        1-1-8-0/eth            CRITICAL
        1-1-7-0/eth            CRITICAL
        1-1-6-0/eth            CRITICAL
        1-1-5-0/eth            CRITICAL
        1-1-4-0/eth            CRITICAL
        1-1-3-0/eth            CRITICAL
        1-1-2-0/eth            CRITICAL
        1-1-1-0/eth            CRITICAL
-----------------------------------------------
```

Manage the MXK-194/198 with Zhone Web Graphical User Interface

The MXK-194/198 device enables Web configuration using the Zhone Web Graphical User Interface (GUI).

Zhone Web GUI

To manage the MXK-194/198 using the Zhone Web GUI:

- Default IP address

  To access the device through the Web Interface Tool through the default device IP address, either access the device through the console port or set the IP address on the management PC to any address in the 192.168.10.0 subnet except 192.168.10.1. The IP address 192.168.10.1 is used as the default address of the Web Interface Tool.

- Configured IP interface

  To access the device through the Web Interface Tool through an IP address that is different from the default IP address/subnet, connect directly to the device using the console port or an accessible Ethernet switch/hub.

  Delete the default autoconfig IP address and add the desired IP address. If the new IP address is not compatible with the address of the management PC, the connection to the device will be lost. Change the address of the management PC to be compatible with the device address.

  Using the Web Interface Tool, configure a default route to the management PC. A default route must be created before the device is accessible across subnets outside the default subnet.

```
zSH> delete ip-interface-record AutoConfig/ip

zSH> interface add 1-1-2-0/eth vlan 94 172.24.94.103/24
Created ip-interface-record ethernet2-94/ip

zSH> route add default 172.24.94.254 1
```

From a PC connected to the 10/100 or GigE port on the MXK-194/198, ensure the IP address on the PC is in the same subnet as the MXK-194/198 IP address. By default, the IP address 192.168.10.1 is assigned to the 10/100 Ethernet port (*1-1-1-0/eth*).

To launch the Zhone Web GUI, in a browser URL address space on the PC enter the IP address configured on the MXK-194/198.

The Zhone Web GUI launches and displays the Login window for the MXK-194/198.

Note: The Web UI supports only Internet Explorer and Mozilla browsers.

Figure 14: Zhone Web GUI Login Screen

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

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

Note: The **del** command can be used to delete all of the Zhone Web User Interface files if needed.

5 INSTALLING THE MXK-194/198

This chapter explains how to install the MXK-194/198 hardware. It includes the following sections:

- Unpack the MXK-194/198 system components, page 105
- Install the chassis into a rack, page 106
- Connect power to the MXK-194/198 and ground the chassis, page 108
- MXK-194/198 system LEDs, page 119
- Connect optical cables, page 122

Note: Before installing the MXK-194/198, read *General safety precautions* on page 33 for important safety and power information.

Unpack the MXK-194/198 system components

Use the following procedure to unpack the MXK-194/198 system components from the shipping cartons.

- On system receipt, check the shipping cartons for physical damage.
- Unpack the shipping cartons, and check the contents for physical damage.
- If the equipment appears damaged, immediately contact the shipping company to file a claim.

The shipping company representative will give instructions on how to submit a claim, where to send the unit, and any special instructions that may be required.

If you need to return the equipment, pack the equipment in its original packing materials and send it by prepaid freight to the address given by the claims representative. If the original packing materials are unavailable, ship the equipment in a sturdy carton, wrapping it with shock-absorbing material.

Install the chassis into a rack

This section explains how to install the MXK-194/198 into a rack:

- Install mounting brackets, page 106
- Mount the system chassis in a rack, page 107

Install mounting brackets

The MXK-194/198 mounting brackets are designed for use in a 19-inch or 23-inch rack.

Installing the mounting brackets onto the MXK-194/198 system chassis

To install the mounting brackets onto the system chassis:

1. Carefully place the system chassis right side up and facing forward on a clean, flat, sturdy work surface.
2. Remove the brackets (2) and screws (12) from their individual packages.
3. Align the bracket so that the rack mount flange is toward the front, centered vertically on the chassis and the 4 screw holes in the chassis align with the 4 screw holes in the bracket.

Note: Use an 8-32 flathead UNC x 0.25 screw when attaching the brackets to the unit. Using the wrong screw type will result in a poorly-secured system. These screws are provided in the installation kit.

4. Secure the two brackets to both sides of the system chassis with the screws provided in the installation kit. See Figure 16 on page 107.

Caution: To prevent damage to the system, use only the screws provided in the installation kit.

Figure 15: Installing rack ears

Mount the system chassis in a rack

The system chassis can be mounted in a 19-inch or 23-inch rack that is connected to an earth ground.

Mounting the MXK-194/198 system chassis in a rack

To mount the system chassis in a rack:

1. Choose a rack position for the system chassis.
2. Carefully lift the system chassis into the rack with the front of the system facing outward.
3. Secure the system chassis to the mounting rack with the screws provided in the installation kit.

Note: Use a 12-24 UNC x 0.5-inch screw when mounting the system to the rack. Using the wrong screw type will result in a poorly-secured system. These screws are provided in the installation kit.

Figure 16: Installing the MXK-194/198 in a rack

Connect power to the MXK-194/198 and ground the chassis

This section explains how to connect power to the MXK-194/198 and ground the chassis:

- Grounding requirements, page 108
- Ground the chassis, page 110
- Connect -48VDC Central Office power to the device, page 111
- Verify the grounding, page 112
- Connect alarms, page 113
- Removable fan trays, page 117

Note: Bare, covered, or insulated grounding conductors must comply with Underwriters Laboratory (UL) standards. Individually covered or insulated grounding conductors shall have a continuous outer finish that is either green or green with one or more yellow stripes. The equipment grounding conductor should be connected to the ground at the service equipment. The grounding cable must be rated at VW-1 or higher.

Zhone recommends grounding the MXK-194/198 using minimum 10 gauge copper wire and NRTL-listed two hole compression-type connectors (such as Amphenol part number 1527272-3).

Grounding requirements

Use the guidelines in this section to provide a system ground for the MXK-194/198.

Before concluding a MXK-194/198 installation and applying DC power, measure the impedance of the building ground reference and ensure that it is less than 1 ohm, for safety. Use an ECOS 1023 POW-R-MATE or an EMC Instrument Model 3710 or similar meter to do this. Zhone recommends that the impedance be 1 ohm or less for proper equipment operation.

If the ground path connected to the MXK-194/198 has an impedance of more than 1 ohm, make improvements to the grounding system before installing the MXK-194/198 equipment.

Other grounding requirements are as follows:

- The earth ground rod is normally buried in the ground at the site. Observe local electrical codes for buried grounding techniques and requirements. Ensure that the ground rod has been installed per local, telco, and NEC code requirements.

- Use a dedicated power source that is only shared with other isolated bonding network (IBN) configured equipment to provide power to the MXK-194/198 and all other related equipment. This configuration prevents interference from possible high surge or noise currents present in some industrial buildings. Otherwise, you must ensure a proper grounding path of less than 1 ohm to the building ground.
- Use the ground bus of a dedicated AC service panel as the location/site ground of the MXK-194/198 equipment. This ground bus must already be connected to the main service panel ground or main building ground reference.
- The impedance of the link between the ground terminal of the MXK-194/198 and the location/site ground to which it is connected must be less than 1 ohm.
- The rack the MXK-194/198 is installed in must be properly grounded.
- Never connect a single-point-ground conductor from the MXK-194/198 to structural steel members or electrical conduits. Specifically, never tie this conductor to a ground source or grounded electrode that is not hard-wired to the building ground reference conductor.
- It is recommended to avoid running in-building cabling near fluorescent lights and other sources of high frequency radiation such as transformers.
- Avoid spliced conductors. Use continuous conductors, which have lower impedance and are more reliable than spliced ones.
- Terminate all conductors in a permanent manner. Ensure all terminations are easily visible and accessible for maintenance purposes.
- Tag ground connections clearly with a message such as “CRITICAL CONNECTION: DO NOT REMOVE OR DISCONNECT.”
- Although some electrical codes permit the use of a conduit as the sole ground conductor between equipment, it is still recommended to use a separate insulated ground conductor through the same conduit. The separate insulated ground conductor maintains the safety ground connection if the conduit is corroded or disconnected.
- Avoid a ground path via serial craft interface RS-232C. The MXK-194/198 RS-232C local craft interface has pins referenced to ground. To prevent undesirable ground path via an attached computer, it is recommended that you only use a portable computer. If only a desktop computer or VT-100 type monitoring equipment is available, use it in conjunction with a UL/CSA Certified RS-232 Opto-Isolator.

Ground conductors for the MXK-194/198 must meet the following requirements:

- No smaller than 10 AWG at any point.
- Does not carry current under normal operating conditions.

- Must be tied to the +48V battery return at the main power Distribution Center
- Should be hard wired to the main ground reference.

Ground the chassis

You must ground the MXK-194/198 chassis before power can be connected to the unit.

Grounding the chassis

1 Place the ground wire onto the grounding screw. The grounding screw is on the lower right corner of the MXK-194/198 rear panel below the grounding symbol (Figure 17 shows the location of the grounding screw. The location is the same for all chassis.)

Figure 17: Grounding the chassis

2 Route a 10 AWG conductor from the chassis to a common 2 AWG frame ground collector that connects to the single point building ground in an IBN. Make sure all ground connections are made with bare metal to bare metal.

Note: For the #8-32 ground stud and hex nuts the recommended torque is 12 to 16 in/lbs.

3 Strip the 10 AWG conductor and crimp a grounding lug to the end of the conductor (Figure 18).

Figure 18: Crimp grounding lug

4 Secure the nuts to the chassis.

5 Connect the ground cable already routed and tighten the bolt.

Tighten the grounding screw to secure the wire.

Note: For the #8-32 ground stud and hex nuts the recommended torque is 12 to 16 in/lbs.

6 The system is now ready to run power.

Connect -48VDC Central Office power to the device

Proper grounding requires that the CO battery return be connected to earth ground.

Power on the MXK-194/198 is located on the rear of the chassis. To provide redundant power, two pairs of positive and negative inputs are located in the power terminals.

Connecting -48VDC Central Office power to the unit

Use the following procedure to connect the wiring between the MXK-194/198 terminal block and the power supplies.

1 Connect the positive wire from power supply A to the terminal marked +. Then connect the negative wire from power supply A to the terminal marked -.

Caution: Use copper wire that is rated for at least two amps of current at 60 VDC.

Figure 19: Redundant DC 48 VDC (dual/redundant inputs on the MXK-194/198)

2 Connect the positive wire from power supply B to the terminal marked +. Then connect the negative wire from power supply B to the terminal marked -.

Caution: Use copper wire that is rated for at least five amps of current at 60 VDC.

3 Connect the other end of the positive (+) wires to Central Office -48 VDC return, and the other end of the negative (-) wires to Central Office -48 VDC.

4 Tighten the screw terminals so that the wires are secure.

Verify the grounding

Verifying proper grounding between the chassis and the rack

Proper grounding reduces the effect of line surges and limits the voltages and RF interference that may affect communication among network devices.

1 Test the impedance from the grounding cable or bar (point 1 in Figure 20) to the rack (point 2 in Figure 20).

The impedance should be less than 1 ohm.

Figure 20: Testing impedance

2 Test the impedance from the MXK-194/198 chassis (point 3 in Figure 20) to the grounding rack.

The impedance must be less than 1 ohm.

The system is now live and ready to provision. For information on the chassis LEDs, see *MXK-194/198 system LEDs* on page 119.

Connect alarms

The MXK-194/198 unit is equipped with input and output alarm terminals. Output alarm terminals provide Central Office-based alarm monitoring when there is a critical alarm or loss of power to the unit. The alarm supports normally closed (N/C), common (COM), and normally open (N/O) contacts, and the alarm relay is rated for one (1) amp at 30 VDC.

Figure 21: MXK-194/198 rear alarm terminals

Connecting output alarms

Use two unshielded wires to connect each unit to N/C or N/O and common.

1 Strip the plastic coating from the wire until approximately .65 cm (1/4 in) of the bare copper is showing.

Figure 22: Alarm outputs

2 Insert the copper end of the one wire into either ALARM N/C or ALARM N/O, depending on the type of alarm (Normally Closed or Normally Open) supported. Connect the other end of the wire to the CO alarm system.

3 Insert the copper end of the other wire into the terminal marked ALARM COM and connect its opposite end to the CO alarm Common terminal.

4 Tighten the screw terminal on the top of each terminal connector.

Table 21: Output alarm pinouts (Pins 5, 6, & 7 of the Power ALM OUT connector)

Pin number	Description
1-4	Used for power, not output alarms
5	NO (Normally Open)
6	Common
7	NC (Normally Closed)

Connecting input alarms

The MXK-194/198 provides an alarm input connector for collecting up to 12 N/O and/or N/C alarms from external devices (e.g. cabinet door open, rectifier fail, fan fail, etc.). The input connector accepts 48-volt inputs directly. All alarm inputs are metallically isolated optocouplers.

48VDC is provided, which is helpful for customer who have “dry” alarm contacts. See Table 22.

Use two unshielded wires to connect each unit to the respective A and B inputs and follow the recommendations below.

1 Strip the plastic coating from the wire until approximately .65 cm (1/4 in) of the bare copper is showing.

Figure 23: Alarm inputs

2 Insert the copper end of the negative voltage (- 48VDC) wire into one of the 12 “B” positions. Connect the other end of the wire to the device N/O or N/C alarm position.

Figure 24: Cabling diagram for alarm inputs

3 Insert the copper end of the positive voltage (48V RTN) wire into the respective “A” position and connect its opposite end to the respective device COM alarm position.

4 After the external N/O and N/C alarms are physically connected, you can then provision the **num2str-profile** on the MXK-194/198 to send an SNMP trap when an alarm condition occurs on the external device. Use the **num2str-profile** to assign a description to an alarm relay. The description will be included in traps and log messages.

The **num2str-profile** uses the following format:

*shelf/slot/***282**/*alarm-contact*

Add a description to the first alarm contact of the MXK-194/198 as follows:

```
zSH> update num2str-profile 1/1/282/1
Please provide the following: [q]uit.
name: -> {Relay 1}: cabinet door open
....................
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

The MXK-194/198 provides pins for unfused -48V supply (-), 48V return (+) and chassis GND on the alarm output connector for external contacts.

Note: When using the unfused -48V power supply provided with the input alarm connector, make sure the unit is turned off when removing and inserting the DB-26 connector. Failing to do so, might cause damages to the alarm circuitry and/or to the unit.

Table 22 lists the input alarms connector pinouts.

Table 22: Alarm input pinouts

Pin		Pin		Pin	
1	(–) BAT -48VDC	10	5A	19	9B
2	1A	11	5B	20	10A
3	1B	12	6A	21	10B
4	2A	13	6B	22	11A
5	2B	14	7A	23	11B
6	3A	15	7B	24	12A
7	3B	16	8A	25	12B
8	4A	17	8B	26	+ 48V RTN/GND
9	4B	18	9A		

Removable fan trays

The MXK-194/198 has a hot swappable fan tray. The tray is replaceable as a complete unit.

WARNING! The MXK-194/198 should not be run for an extended time without the fan tray. The fan tray should only be removed when replacing with another fan tray.

Removing the fan tray

1 Turn the unlock screw counter clockwise with a screwdriver to unlock the fan tray.

2 Pull the handle to partially remove the fan tray but do not remove it completely until after the fans have stopped rotating.

Inserting the fan tray

1 To insert the fan tray, line it up and slide it in.

2 Turn the unlock screw clockwise with a screwdriver to lock the fan tray.

MXK-194/198 system LEDs

The MXK-194/198 system LEDs are located on the front bezel, as shown in Figure 25.

Figure 25: MXK-194/198 LEDs

Table 23 describes the MXK-194/198 system LEDs.

Table 23: MXK-194/198 front LED descriptions

LED	Description
Power (green)	ON: Voltage is within tolerance. OFF: Voltage is not operational.
Diag/Fault (orange)	ON: During power up, and when there is an alarm on the unit. OFF: Unit is operating normally.
Operational (green)	BLINKING: Unit is booting up. ON: Device is operating normally.

Note: A link down alarm will be present for any Ethernet uplink ports that are vacant or not linked, resulting in the diag LED becoming illuminated.

Management interface LEDs

Figure 26: Management interface port and LEDs

Table 24: Management interface LEDs

LED	Color/Behavior	Description
Link	Green/On	Link is active
	Off	No link
Activity	Yellow/Blinking	Activity on link
	Off	No activity

10GE interface LEDs

Figure 27: 10GE interfaces ports and LEDs

Table 25: 10GE interface LEDs

LED	Color/Behavior	Description
Link	Green/On	Link is active
	Off	No link
Activity	Yellow/Blinking	Activity on link
	Off	No activity

FE/GE interface LEDs

Figure 28: FE/GE interfaces ports and LEDs

Table 26: FE/GE interfaces LEDs

LED	Color/Behavior	Description
Link	Green/On	Link is active
	Off	No link
Activity	Yellow/Blinking	Activity on link
	Off	No activity

GPON interface LEDs

Figure 29: GPON interfaces ports and LEDs

Table 27: GPON interfaces LEDs

LED	Color/Behavior	Description
Link	Green/On	Link is active
	Off	No link
Activity	Yellow/Blinking	Activity on link
	Off	No activity

Connect optical cables

This section explains how to use optical cables with the MXK-194/198:

- Fiber connections, page 122
- Fiber management, page 122
- Fiber guidelines and installation, page 122

Fiber connections

Optical cables must be properly routed to the MXK-194/198.

Before making any connections, be sure that optical cables and components are clean and free of dust and debris. For cleaning information, see *Selecting cleaning materials* on page 49for more information.

When making a fiber optic connection, avoid touching the fiber cable ends to the outside of the mating connector. Touching can contaminate the connectors.

The MXK-194/198 uses the following optical fibers:

- Multimode fiber for connections using 850 nm
- Single-mode fiber for connections using 1310 nm or 1550 nm

Fiber management

Zhone Technologies provides a fiber management solution for the MXK-194/198. When fiber management installations are properly planned, it allows for service or replacement of any removable subassembly contained in an equipment shelf without having to remove or alter the cable harness support structure used during the original installation. Long-term system maintenance and system growth should be considered when installing or cabling Zhone systems.

Fiber guidelines and installation

Fiber Cable Bend Radius: A minimum bend radius of 30 mm is recommended to guarantee the specified system performance. Sharp bends in fiber cables create undesirable optical attenuation or loss.

Grouping Fiber Cables: Flexible spiral wrap conduit in various sizes is recommended to control fiber bundles. Cable ties are not recommended for securing or bundling fiber cables. The spiral wrap conduit may be secured using cable ties as long as the cable tie cinch force does not crush the conduit.

Bundling copper cables and fiber cables separately is recommended because cable tie cinch force used to retain heavier copper cables will often distort mixed-in fiber cables generating unpredictable optical transmission power losses.

6 GPON ON THE MXK-194/198

This section includes:

- GPON terminology, page 123
- Bridge add, interface add, host add commands in Smart OMCI and Dynamic OMCI, page 126
- Planning GPON networks, page 127
- Installation testing, page 129

Although this guide is primarily concerned with configuring Zhone equipment, it is also important to have a strong understanding of the underlying technology. This section defines some general concepts to consider, and is not to be a definitive resource.

Note: All the commands that start with **gpononu** or **gponolt** can be replaced to start with **onu** or **olt**. For example: > **gpononu set** is same as > **onu set** ; > **gponolt show bw** is same as > **olt show bw**.

GPON terminology

This section describes:

- Components of GPON optical deployment networks, page 123.
- Relationship between T-conts and GEM ports, page 124.

Components of GPON optical deployment networks

Optical networks are comprised of a number of components between the subscriber devices.

- OLT

  Optical Line Terminator. This device is considered the head end of the ODN. (Note that each port on a GPON line card is considered an OLT.)

- Optical fiber

  The optical fiber is the physical cable.

- Optical splitters (GPON only)

Optical splitters split a single optical signal to multiple optical signals.

- Couplers

  Couplers are connectorized means for splicing cables. Because couplers are connectors there is an optical signal cost for connectors

- ONT or ONU

  Optical Network Terminator (ONT) and Optical Network Unit (ONU) are reasonably similar terms which are both defined in the ITU-T G.984 GPON standards. They both provide an end for the ODN and conversion to some electrical media; However, ONTs usually have multiple subscriber-side services and interfaces, like Ethernet LAN, POTS or coaxial cable for TV services. ONUs would have a GPON interface uplink (just like the ONT), but in the downstream direction they provide last mile copper access ports such as a VDSL2 or Fast Ethernet which connects to customer premises equipment such as a VDSL2 modem or an Ethernet Hub, Switch or Residential Gateway.

- Attenuators

  Attenuation is the term for the loss of optical power on the ODN. Some devices may actually receive too high a signal power strength for the receiving device. This situation most commonly occurs in lab settings. An attenuator can adjust the power strength of the optical signal.

All the fiber components named above are important in planning and installing GPON networks.

Relationship between T-conts and GEM ports

Figure 30 shows the relationship between T-conts and GEM ports.

Figure 30: Relationship between T-conts and GEM ports

- T-Conts

  Transmission Container (T-cont)s are how the ONU represents a group of logical connections that appear as a single traffic-bearing entity for the purpose of bandwidth assignment on the upstream side of the ONU.

  Each ONU contains one or more T-conts. The OLT discovers the number of T-conts supported by a given ONU and assigns Alloc IDs to T-conts in this ONU. Alloc ID is the identifier of a T-cont.

  Each T-cont contains one or more GEM ports. The Alloc ID is assigned to a T-cont during the GEM port creation.

  Bandwidth allocation on a T-cont is defined in the GPON traffic profile. Multiple GEM ports can share one T-cont by enabling shared feature in the associated GPON traffic profile.

- GEM ports

  GPON Encapsulation Method (GEM) ports are how the ONUs separate the services from the upstream side of the ONU to the downstream ports. Each of these GEM ports needs to be unique on the ODN for the OLT port.

GEM port ID is the identifier of a GEM port. There are two types of GEM port IDs, Dynamic GEM port IDs used in the Smart OMCI provisioning and Arbitrary GEM port IDs used in the Dynamic OMCI provisioning.

GEM ports are dynamically created during the **bridge add, interface add** and **host add** operations. Conversely, GEM ports can be automatically deleted during the **bridge delete**, **interface delete** and **host delete** operations.

The traffic shaping on a GEM port is defined in a CPE traffic management Profile.

Note: When creating a GEM port, a GPON traffic profile (that defines T-cont) must be specified.

For detailed configurations and additional information on GEM ports, refer to the following sections in *Chapter 11, GPON Subscriber Interfaces*:

Dynamic GEM ports on page 333 (Smart OMCI provisioning)

Arbitrary GEM ports on page 371 (Dynamic OMCI provisioning)

GEM port creation on page 484

Bridge add, interface add, host add commands in Smart OMCI and Dynamic OMCI

The **bridge add, interface add,** and **host add** commands define the transport type, port and interface in the SLMS device by the *shelf-slot-port-subport (or interface)/transport type* syntax.

For the MXK-194/198, since there are no cards, *slot* is fixed at *1*. *Port* is the physical port. *Subport* may be different depending on the *transport type*.

Based on the methods of GPON ONTs provision, the syntax of the **bridge add, interface add, host add** command are different (the following two examples assume the GPON traffic profiles 1, 2, and 3 exist):

- If this ONU is managed by Smart OMCI:

```
zSH> bridge add 1-1-4-701/gponport gtp 1 downlink vlan 101
Adding bridge on 1-1-4-701/gponport
Created bridge-interface-record 1-1-4-701-gponport/bridge

zSH> interface add 1-1-1-501/gponport gtp 2 vlan 209 172.24.1.1/24
Created ip-interface-record 1-1-1-501-209/ip.

zSH> host add 1-1-6-901/gponport gtp 3 vlan 500 static 192.168.49.2
Adding host for 1-1-6-901/gponport
```

- If this ONU is managed by Dynamic OMCI:

The **bridge add** command defines the bridge transport type, port and interface by the *shelf-slot-OLT port-ONU port*/**gpononu gem** *GEM port* syntax.

```
zSH> bridge add 1-1-4-1/gpononu gem 701 gtp 1 downlink vlan 101
Adding bridge on 1-1-4-1/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge

zSH> interface add 1-1-1-1/gpononu gem 501 gtp 2 vlan 209 172.24.1.1/24
Created ip-interface-record 1-1-1-501-209/ip.

zSH> host add 1-1-6-1/gpononu gem 567 gtp 3 vlan 500 static 192.168.49.2
Adding host for 1-1-6-1/gpononu
```

Planning GPON networks

When deploying GPON networks, you have to think in optical terms, rather than electrical or copper based terms. With copper based solutions you think of distance and transport technology (“Will ADSL or VDSL reach from the CO to the subscribers?” is a significant network design question); with fiber based networks, and GPON in particular, you have to think in terms of optical link power loss budgets.

Link loss is the amount of signal attenuation as you proceed farther away from the OLT toward the subscribers’ ONTs. Each component, including the fiber cable itself, degrades the signal. Attenuation is the term used for describing the amount of signal degradation.

Figure 31: Link loss in an GPON Optical Deployment Network

The plan for both a GPON network and Active Ethernet network should include a link loss budget map that shows how each component, even the distance of each length of fiber, should affect signal attenuation. Because GPON lines are split into multiple lines which have a significant power loss, the link loss budget map is a more important requirement for GPON.

Note: The power loss may vary by manufacturer, refer to equipment vendor for the detail.

Component	Loss
Optical fiber	-0.3 dB per kilometer
Splitters	The link loss for splitters depends on the number of splits • 2 splits, -4 dB • 4 splits, -7.5 dB • 8 splits, -11 dB • 16 splits, -14 dB • 32 splits, -18 dB • 64 splits, -21.5 dB
Splices	-0.1 dB
Connectors	-0.2 dB

Component	Loss
Couplers	Couplers are connectorized means for splicing cable. -0.4 dB

Installation testing

The theoretical link loss budget map is very important when installing fiber. Testing should be done before and after each component is added. Matching the actual signal attenuation with the theoretical link loss budget map helps identify problems such as

- macro bends in cables (too small a bend radius)
- connector loss from back reflection (the contact between the face ends of fiber in a connector, or a splice)
- incorrectly matching UPC and APC connectors may also create back reflections. UPC connectors (Ultra Physical Contact) have a slightly spherical end face. APC connectors (Angled Physical Contact) use an industry standard angle on the end face of the fiber. (Though you should be aware of older, non standard APC connectors which use a different angle.)

Figure 32: End face of UPC and APC connectors

There are testing tools on the market which can be used to test the components as added.

The actual figures that are discovered during installation testing should also be noted and filed as they may also be helpful when troubleshooting problems which may arise in the ODN in the future.

7 IP CONFIGURATION

This chapter explains IP services on the MXK-194/198. It includes the following sections:

- Overview, page 131
- Terminology and concepts, page 133
- Routing types: host–based and network–based, page 135
- Routing and IP addresses, page 139
- IP services, page 141
- IP provisioning examples, page 156
- IP administrative procedures, page 192

This chapter does not include giving the MXK-194/198 an IP address for management purposes. To give the MXK-194/198 an IP address for out of band management, see Configure a management interface, page 57

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 based on MAC address (ISO Logical Link layer 2, bridging)
- Packets are delivered based on IP address (ISO Network layer 3, routing)

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 layers referred to above are part of the Open Systems Interconnection (OSI) reference model as shown in Table 28. While not all protocols follow

the OSI model, the OSI model is helpful for understanding variations of network functionality.

Table 28: 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. The response follows the same process as shown in Figure 33.

Figure 33: Applications requested networked information

Layer 3, the network layer, handles the delivery of data packets from source to destination. Any device connected to a network is considered a host or a node on that network. Zhone devices with IP capability can act as routers to accept network traffic and forward it on to host destinations based on IP addresses. To get from source to destination, the IP packet passes through many nodes, or hops, along the way.

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 contains the best route to a given destination.

All routers maintain routing tables of the sequence of hops taken from source to destination. The routing table is used by the router to direct datagrams most efficiently. The routing table information is also shared with other routers on the same network.

Bridges direct frames based on MAC addresses. Every device on the Internet has a unique MAC address. IP addresses may be given out dynamically as needed, so at times the device may not have an IP address.

Terminology and concepts

This section discusses routing terminology and concepts:

- Physical port, page 133
- Physical interface, page 134
- Logical interface, page 134
- Numbered and unnumbered interfaces, floating interfaces, page 135

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.

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-194/198 for bridging and routing, separate VLANs must be used.

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 25 (actually 24 pairs = 50 pins) 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.

Figure 34: 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, such as using GEM ports to separate a single physical port into multiple virtual ports on a GPON link to an ONU.
- A collection of physical ports. Bonded links or IMA groups use multiple physical ports as one physical 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 for ADSL or SHDSL), or both.

IP interfaces are either a logical interface directly given to a physical interface (**interface add**), or a logical interface indirectly given to one or more physical interfaces (**interface add float** and **host add commands**).

For information about bridging interfaces, see Bridging Configuration, page 203.

Numbered and unnumbered interfaces, floating interfaces

A numbered interface has an IP address given to the physical interface. Numbered interfaces are given using the **interface add** command. An unnumbered interface (also called a floating interface) is the mechanism to give an IP address, not to a physical interface, but a floating logical interface which is associated with the system. The floating IP address is created using the **interface add float** command. This floating interface may then be shared with physical interfaces using the **host add** command. Please see *Routing types: host–based and network–based* on page 135.

Floating interfaces are similar to Cisco loopback interfaces.

Numbered and unnumbered interfaces, floating interfaces

A numbered interface has an IP address given to the physical interface. Numbered interfaces are given using the **interface add** command. An unnumbered interface (also called a floating interface) is the mechanism to give an IP address, not to a physical interface, but a floating logical interface which is associated with the system. The floating IP address is created using the **interface add float** command. This floating interface may then be shared with physical interfaces using the **host add** command. Please see *Routing types: host–based and network–based* on page 135.

floating interfaces are similar to Cisco loopback interfaces.

Routing types: host–based and network–based

Zhone supports two routing types:

- *Network-based (numbered) routing overview* on page 136
- *Host-based (unnumbered) routing overview* on page 137

Network-based routing allows a single routing table entry to represent many numbered host addresses. A numbered IP address on the interface is required. The subnet is defined by the numbered IP address.

Network-based routes are commonly used in situations where you have a large number of devices per interface which can be in the same subnet.

Figure 35: With network-based routing, the IP address is on the physical interface

Host-based routing allows for a granular allocation of addresses based on the host floating IP address. Host-based routing uses floating IP addresses; that is, the IP address may be for more than one physical interface, or in other words, you may have devices on different interfaces which belong to the same subnet.

Figure 36: With host-based the IP address is not on a physical interface and may be associated to multiple physical interfaces. This association means devices on different physical ports may be in the same subnet.

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 the 10.10.10.0 subnet. All 254 addresses in this subnet are assigned to this interface, regardless of how many addresses in this subnet are actually used.

Unlike host-based routing, network based-routing requires numbered IP interfaces on the MXK-194/198 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-194/198 numbered interface.

See *Network-based routing* on page 156 for network-based configuration procedures. Table 29 describes host-based and network-based commands for adding IP interfaces.

Table 29: 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	Single per **host add** command	unnumbered
Interface add	Network-based routing with bridge or router	Static	Multiple based on subnet mask length	numbered

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, a 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-194/198 are on the same subnet as the MXK-194/198 floating interface.

See *Host-based routing* on page 167 for host-based routing configuration procedures.

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 30 describes the default routing preferences on the device. These preferences cannot be overridden. Higher numbers indicate more preferred route types:

Table 30: Routing preferences

Type of route	Default preference
Local	10
Static	9
RIP	4
Static low (used for default routes)	4

Routing and IP addresses

Routers direct packets based on IP address information. Routing protocols determine the path the packets follow when traversing from one host to another.

The MXK-194/198 and other Zhone SLMS devices

- may provide addresses dynamically
- may be relay agents for other DHCP servers.

MXK-194/198s also support static IP addresses.

IP addresses for downstream devices

Devices which are downstream from the MXK-194/198 may obtain an IP address from the MXK-194/198 or with the MXK-194/198 as a relay agent (see Figure 37).

- The MXK-194/198 is a DHCP server

  See *MXK-194/198 DHCP server support* on page 146

- Another device is a DHCP server and the MXK-194/198 is a DHCP relay agent.

  See *MXK-194/198 DHCP relay* on page 150

- The downstream interface is given a static IP address

  See *Static network-based routing (without DHCP)* on page 157

Figure 37: The MXK-194/198 may provide IP addresses for downstream devices

devices request
IP address; IP address
given by MXK-19x
DHCP
Server
MXK-19x is DHCP server
devices request
IP address; IP address
given by upstream
device: DHCP relay
information added
in messages
DHCP
Server
Other device is
DHCP server
MXK-19x is DHCP relay agent
Downstream device has static
IP address or from downstream
device with DHCP
IP is assigned statically to IP interface

IP services

The MXK-194/198 provides the following IP services:

- IP forwarding and routing

  Incoming packets from an interface are forwarded to the appropriate output interface using the routing table rules.

- Domain Name System (DNS)

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

  See Configuring DNS resolver, page 143.

- Dynamic Host Control Protocol (DHCP) servers simplify user IP address configuration.

  The MXK-194/198 may act as a local DHCP server. DHCP is the means for dynamically assigning IP addresses. Basically, a DHCP server has a pool of IP addresses which 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.

  See MXK-194/198 DHCP server support, page 146.

- DHCP relay provides access to upstream DHCP servers

  The MXK-194/198 may also act as a DHCP relay agent, supporting DHCP requests from downstream devices to upstream DHCP servers. The MXK-194/198 supports primary and alternate DHCP server configurations. DHCP relay supports Option 82 insertion to identify the requesting client to the DHCP server.

  See MXK-194/198 DHCP relay, page 150.

- IP fallback/IP redundancy

  The MXK-194/198 supports IP redundancy which may also be called 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.

  See IP fallback route, page 150.

- Routing Information Protocol (RIP)

  RIP is an interior gateway protocol (IGP) which is widely used for routing traffic on the Internet. RIP performs routing within a single autonomous system based on distance-vector algorithms which 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.

Routing Information Protocol version 2 (RIPv2) is an enhancement to RIP. RIPv2 allows more information to be included in RIP packets and provides an authentication mechanism.

RIPv1 is classful, supporting the five IPv4 classes: A, B, C, D, E. RIPv2 supports the Classless Inter-Domain Routing (CIDR) routing scheme which uses the address space aggregation method. CIDR addresses set up a subnet using a slash to define the subnet (and hence the netmask). For example the 10.10.10.0 subnet with subnet mask 255.255.255.0, can be shown as 10.10.10.0/24. The 24 refers to the first three eight bit groupings (hence 24 bits) of the network address. So the last three eight bit groupings provides 254 addresses in the subnet.

See RIP configuration, page 152.

- IP TOS/COS support

  The MXK-194/198 supports the marking and remarking of TOS values in IP packets and COS values in Ethernet VLAN headers as defined by IETF RFC1349 and IEEE 802.1p respectively. The configured TOS and COS levels specify the packet priority and queueing methods used to transport the packet through the IP and Ethernet networks. The MXK-194/198 sets and transports the TOS/COS values, while the switches and routers connected to the MXK-194/198 perform the queuing services and packet QOS processing.

  See ToS, CoS, and sCoS on an IP interface, page 153.

- IP Service Level Agreement (IPSLA)

  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.

  See IP Service Level Agreement (IPSLA), page 517.

Configuring 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. DNS resolver is used internally to the SLMS device, not as a service for other devices.

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.

The **resolver** profile supports the following parameters (all others should be left at their default values):

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.
```

Note: DNS resolver is a system wide service. Only one DNS resolver may be configured for the MXK-194/198 system.

Optionally, you can create a **hosts** profile after the **resolver** profile has been created. The syntax is **new host-name** *routingdomain*/*ipoctet1*/*ipoctet2*/*ipoctet3*/*ipoctet4*.

The **host-name** profile supports the following parameters (all others should be left at their default values):

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.
```

DHCP

Dynamic Host Control Protocol (DHCP) is the means for dynamically assigning 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, as well as the IP address from the DHCP server.

When the MXK-194/198 acts as a local DHCP server, the MXK-194/198 can assign temporary (leased) IP addresses to clients. Each DHCP client sends a request to the MXK-194/198 for an IP address lease. The MXK-194/198 then assigns an IP address and lease time to the client. The MXK-194/198 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-194/198 attempts to assign the same IP address to the same client on DHCP lease renewal.

The MXK-194/198 allows for shared pools of IP addresses. With shared pools supported both on numbered and floating interfaces — ranges of IP addresses can be given to a subnet from a numbered interface (single physical interface) or floating interface (multiple physical interfaces).

The MXK-194/198 may also act as a DHCP relay agent, supporting DHCP requests from downstream devices to upstream DHCP servers. The MXK-194/198 supports primary and alternate DHCP server configurations. DHCP relay supports Option 82 insertion to identify the requesting client and other information to the DHCP server.

This section discusses:

- MXK-194/198 DHCP server support, page 146
- DHCP server profiles and scope, page 146
- DHCP server options, page 147
- DHCP server subnet options, page 148
- MXK-194/198 DHCP relay, page 150

MXK-194/198 DHCP server support

The MXK-194/198 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-194/198 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-194/198 uses the following profiles to configure DHCP servers:

- **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 page 147 for more information.
- **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 page 148 for more information.

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** (per host)
2. **dhcp-server-group** (per group of hosts within a subnet)
3. **dhcp-server-subnet** (per subnet)
4. **dhcp-server-options** (per system)

If a parameter is set in multiple profiles (for example, lease times or default routers), the MXK-194/198 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-194/198 uses those settings as the default for all DHCP server interfaces. For information about logging DHCP requests, see *DHCP logging* on page 196.

DHCP server options

At startup, the MXK-194/198 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
dhcp-server-options  0
lease-time: ----->  {43200}
min-lease-time: ->  {0}
max-lease-time: ->  {86400}
reserve-start: -->  {1}
reserve-end: ---->  {1}
restart: -------->  {no}
```

Table 31 describes the **dhcp-server-options** profile that supports the following configurable parameters (all others should be left at their default values):

Table 31: 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-194/198 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-194/198 ‘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.
restart	Used to bounce the DHCP server.

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 32 describes the **dhcp-server-subnet** profile that supports the following configurable parameters (all others should be left at their default values):

Table 32: 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. Ranges cannot overlap.
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. Ranges cannot overlap.
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. default: -1 The values of the DHCP server options profile are used.
max-lease-time	See description in **dhcp-server-options** profile. default: -1 The values of the DHCP server options profile are used.
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. For host-based, the floating IP address. For network-based, the IP address of the interface.
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.

Table 32: dhcp-server-subnet profile configurable parameters (Continued)

Parameter	Description
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.
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-subnet 1
network:--------------->   {10.107.8.0} references floating IP address
netmask:--------------->   {255.255.255.0}sets 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:---->   {86400} lease times (86,400 seconds is one day)
min-lease-time:-------->   {86400} lease times
max-lease-time:-------->   {259200} lease times (259,200 seconds is three days)
boot-server:----------->   {} for bootp services
bootfile:-------------->   {} for bootp services
default-router:-------->   {10.107.8.254}
primary-name-server:--->   {172.16.160.250} IP address for DNS server
secondary-name-server:->   {172.8.120.250} 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-194/198 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. If multiple floating IP records are present, the desired <name>/<type> may be specified.

In DHCP relay configurations, the MXK-194/198 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-194/198 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-194/198 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.

IP fallback route

The MXK-194/198 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 use to determine route availability. The MXK-194/198 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 fallback route

To configure IP fallback route:

1. Add a route with the IP addresses of the nexthop router and fallback router.

```
zSH> route add default 192.168.34.254 1 fallback 192.168.34.201 2000 3

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  192.168.34.201
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. To 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-194/198 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-194/198:

1. Enable RIP for the system as a whole.

```
zSH> rip enable
```

2. View the available IP interfaces.

```
zSH> interface show
3 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 192.168.254.159/24  00:01:47:27:14:59       1-1-1-0-eth
1/1/2/0/ip     DOWN    1 10.250.1.254/24     00:01:47:27:14:5b      1-1-2-0-eth-100
1/1/3/0/ip     DOWN    1 192.168.1.10/24     00:01:47:27:14:5a       1-1-3-0-eth-55
-------------------------------------------------------------------------------
```

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 1-1-2-0-eth-100 listen v1v2
zSH> rip interface 1-1-3-0-eth-55 listen v1v2
```

4. To enable transmission of RIP advertisements on an interface:
   a. `zSH> rip interface 1-1-3-0-eth-55 talk v2`

      or

   b. `zSH> rip interface 1-1-3-0-eth-55 talk v1compat`

ToS, CoS, and sCoS on an IP interface

This section discusses these topics:

- IP Quality of Service (QoS), page 153
- Fields in IP header, page 153
- 802.1p priority queues, page 154
- Fields in the VLAN header, page 154
- ToS, CoS, sCoS parameters, page 154

IP Quality of Service (QoS)

The MXK-194/198 supports IP QoS. This service provides the ability to assign a service level or Type of Service (ToS) to an IP interface. The ToS service level specifies the packet priority and queueing methods used to transport the packet through the IP network.

The MXK-194/198 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 the packet priority and queueing methods used to transport the packet through the IP and Ethernet networks. The MXK-194/198 sets and transports the ToS/CoS values, while the switches and routers connected to the MXK-194/198 perform the queuing services and packet QoS processing.

The MXK-194/198 originates and preserves the ToS settings to ensure these settings are passed to other IP devices in the network.

Note: ToS bits are not altered for VoIP Real Time Transport Protocol (RTP) packets, which have their own ToS bit settings set in the **voip-server-entry** profile regardless of the ToS setting on the outgoing interface.

This service enables you to:

- Add IP packet ToS values and VLAN header CoS values to packets originating from the MXK-194/198.
- Overwrite existing IP packet ToS values and VLAN header CoS values that are transported through the MXK-194/198.
- Leave existing IP packet ToS values and VLAN header CoS values unchanged in all packets.

Fields in 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 or Class of Service (CoS) value based on eight (0-7) levels of service. This field contains information about how the traffic should be forwarded. The MXK-194/198 supports basic ToS marking without queue servicing options in the **ip-interface-record** profile.

802.1p priority queues

Multi-media Traffic Management (MTM), is a rules-based policy enforcement mechanism for SLMS systems. The MXK-194/198 MTM is used to mark packet priorities and service queues. The MXK-194/198 supports four 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-194/198's implementation of the MTM feature set for QoS.

Fields in the 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-194/198 supports basic CoS marking and remarking without any queue servicing options.

ToS, CoS, sCoS parameters

Table 33 describes the parameters in the **ip-interface-record** profile used for ToS and CoS support.

Table 33: 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-194/198 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.
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}
reasmmaxsize:---------------->   {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}
```

IP provisioning examples

This section describes the following procedures:

- Network-based routing, page 156
- Host-based routing, page 167
- Host-based routing for data and voice services on GPON, page 189

Network-based routing

Network-based routing assigns one IP to the interface and the entire subnet represented by that one address in a single routing table entry. The subnet masks can be of variable lengths.

For an overview of network-based routing see Network-based (numbered) routing overview, page 136.

You can configure network-based routing on the MXK-194/198 in one of three ways:

- configuration without a DHCP server.

  See *Static network-based routing (without DHCP)* on page 157

- DHCP services are on the MXK-194/198 (the MXK-194/198 is the DHCP server).

  Network-based routing with the MXK-194/198 as local DHCP server on page 160

- The MXK-194/198 as a DHCP relay agent for an external DHCP server.

  Network-based routing with an external DHCP server on page 164

Static network-based routing (without DHCP)

Network-based routing supports creating a static route on a numbered interface. Figure 38 shows a static route example.

Figure 38: Static routing example

Configure a static network-based route

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.

1. Create an IP interface on an Ethernet uplink port for the upstream connection.

```
zSH> interface add 1-1-3-0/eth 192.169.1.14/24
Created ip-interface-record 1-1-3-0-eth/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
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 10.55.1.198/24      00:01:47:24:08:ae       1-1-1-0-eth
1/1/3/0/ip     UP      1 192.169.1.14/24     00:01:47:24:08:b1       1-1-3-0-eth
-------------------------------------------------------------------------------
```

2. Create GPON traffic profile.

```
zSH> new gpon-traffic-profile 1
gpon-traffic-profile  1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

3 Create an IP interface on a GPON GEM port.

```
zSH> interface add 1-1-1-1/gpononu gem 501 gtp 1 vlan 209 172.24.1.1/24
Created ip-interface-record 1-1-1-501-209/ip.
```

GEM port 501 is created on ONU 1/1/1 as well.

4 Verify the interface.

```
zSH> interface show
3 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
--------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 10.55.1.198/24      00:01:47:24:08:ae       1-1-1-0-eth
1/1/1/501/ip   UP      1 172.24.1.1/24       00:01:47:24:08:af       1-1-1-501-209
1/1/3/0/ip     UP      1 192.169.1.14/24     00:01:47:24:08:b1       1-1-3-0-eth
--------------------------------------------------------------------------------
```

5 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 1-1-1-501-209
ip-interface-record  1-1-1-501-209/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: ---------------------->  {209}
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 GPON GEM port.

```
zSH> interface delete 1-1-1-501/gponport
Delete complete
```

Verify the IP interface is deleted.

```
zSH> interface show
2 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
--------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 10.55.1.198/24      00:01:47:24:08:ae       1-1-1-0-eth
1/1/3/0/ip     UP      1 192.169.1.14/24     00:01:47:24:08:b1       1-1-3-0-eth
--------------------------------------------------------------------------------
```

Verify the **ip-interface-record** profile is deleted from the system.

```
zSH> list ip-interface-record
ip-interface-record  1-1-1-0-eth/ip
ip-interface-record  1-1-3-0-eth/ip
2 entries found.
```

Network-based routing with the MXK-194/198 as local DHCP server

You can configure the MXK-194/198 to act as a local DHCP server in a network-based routing configuration. Figure 39 shows network-based routing with the MXK-194/198 as DHCP server.

Figure 39: Network-based routing with MXK-194/198 as DHCP Server

Configuring network-based routing with MXK-194/198 as local DHCP server

Specifying *server* in the CLI enables the DHCP server functionality locally on the MXK-194/198. However, services such as DNS or bootp are not enabled.

1 Create an IP interface on an Ethernet uplink port.

```
zSH> interface add 1-1-4-0/eth vlan 777 192.169.1.14/24
Created ip-interface-record 1-1-4-0-eth-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
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 172.24.200.68/24    00:01:47:27:14:54       1-1-1-0-eth
1/1/4/0/ip     DOWN    1 192.169.1.14/24     00:01:47:27:14:55   1-1-4-0-eth-777
-------------------------------------------------------------------------------
```

2 Create GPON traffic profile.

```
zSH> new gpon-traffic-profile 1
gpon-traffic-profile  1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

3 Create the IP interface on a GPON GEM port.

```
zSH> interface add 1-1-5-1/gpononu gem 601 gtp 1 vlan 777 172.24.1.1/24
server
Created ip-interface-record 1-1-1-601-777/ip.
```

The **ip-interface-record** profile is created with the DHCP server functionality enabled. See Step 5.

4 Verify the interface.

```
zSH> interface show
3 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 172.24.200.68/24    00:01:47:27:14:54       1-1-1-0-eth
1/1/4/0/ip     DOWN    1 192.169.1.14/24     00:01:47:27:14:55       1-1-4-0-eth-777
1/1/5/601/ip   DOWN    1 172.24.1.1/24       00:01:47:24:08:af       1-1-5-601-777
-------------------------------------------------------------------------------
```

5 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 1-1-5-601-777
ip-interface-record  1-1-5-601-777/ip
vpi: ------------------------->  {0}
vci: ------------------------->  {0}
rdindex: --------------------->  {1}
dhcp: ------------------------->  {server}<--------
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: ------------>  {true}<--------
subnetgroup: ----------------->  {0}
unnumberedindex: ------------->  {0}
mcastcontrollist: ------------>  {}
vlanid: ---------------------->  {777}
maxVideoStreams: ------------->  {0}
tosOption: ------------------->  {disable}
tosCOS: ---------------------->  {0}
vlanCOS: --------------------->  {0}
s-tagTPID: ------------------->  {0x8100}
s-tagId: --------------------->  {0}
s-tagIdCOS: ------------------>  {0}
```

6 Add a floating IP interface for the dhcp-server-subnet.

```
zSH> interface add float flt 10.1.1.1/24
Created ip-interface-record flt/ip.
```

7 Create a **dhcp-server-subnet** profile for the floating interface.

```
zSH> new dhcp-server-subnet 1
dhcp-server-subnet  1
Please provide the following: [q]uit.
network: --------------->  {0.0.0.0}: 10.1.1.0
netmask: --------------->  {0.0.0.0}: 255.255.255.0
domain: ---------------->  {0}:
range1-start: ---------->  {0.0.0.0}: 10.1.1.2
range1-end: ------------>  {0.0.0.0}: 10.1.1.254
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}: 10.1.1.1
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.
```

8 Add the **dhcp-server-subnet** group number to the **ip-interface-record** of the DHCP server.

```
zSH> update ip-interface-record 1-1-5-601-777/ip
ip-interface-record  1-1-5-601-777/ip
Please provide the following: [q]uit.
vpi: ------------------------->  {0}:
vci: ------------------------->  {0}:
rdindex: --------------------->  {1}:
dhcp: ------------------------>  {server}:  ** read-only **
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: ------------>  {true}:
subnetgroup: ----------------->  {0}: 1
unnumberedindex: ------------->  {0}:
mcastcontrollist: ------------>  {}:
vlanid: ---------------------->  {777}:
maxVideoStreams: ------------->  {0}:
tosOption: ------------------->  {disable}:
tosCOS: ---------------------->  {0}:
vlanCOS: --------------------->  {0}:
s-tagTPID: ------------------->  {0x8100}:
s-tagId: --------------------->  {0}:
s-tagIdCOS: ------------------>  {0}:
....................
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Deleting the configuration

When necessary, delete the GEM port. The IP interface on the GEM port is deleted as well.

```
zSH> interface delete 1-1-5-601/gponport
Delete complete
```

The **ip-interface-record** profile is deleted from the system.

Network-based routing with an external DHCP server

The MXK-194/198 acts as a DHCP relay agent to an external DHCP server. Figure 40 shows network-based routing with an external DHCP server.

Figure 40: Network-based routing with external DHCP Server

Configuring network-based routing with external DHCP server

1 Create an IP interface on an Ethernet uplink port.

```
zSH> interface add 1-1-4-0/eth vlan 777 192.169.1.14/24
Created ip-interface-record 1-1-4-0-eth-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
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 172.24.200.68/24    00:01:47:27:14:54       1-1-1-0-eth
1/1/4/0/ip     DOWN    1 192.169.1.14/24     00:01:47:27:14:55       1-1-4-0-eth-777
-------------------------------------------------------------------------------
```

2 Create GPON traffic profile.

```
zSH> new gpon-traffic-profile 1
gpon-traffic-profile  1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

3 Create an IP interface on a GPON GEM port.

```
zSH> interface add 1-1-7-1/gpononu gem 701 gtp 1 vlan 777 10.109.8.1/24
Created ip-interface-record 1-1-7-701-777/ip.
```

4 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 1-1-7-701-777/ip
Created DHCP Relay Agent: group: 1, index: 1
```

5 View the **dhcp-server-subnet** profile.

```
zSH> get dhcp-server-subnet 1
dhcp-server-subnet  1
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: ----------->  {1}
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 1
Deleted DHCP Relay Agent group: 1, index: 1
```

The **dhcp-server-subnet** 1 profile is deleted.

2 Delete the GEM port. The IP interface on this GEM port will be deleted as well.

```
zSH> interface delete 1-1-7-701/gponport
Delete complete
```

Host-based routing

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.

You can configure host-based routing on the MXK-194/198 in one of three ways:

- Static configuration without a DHCP server.

  See *Static host-based routing (without DHCP)* on page 168

- DHCP services are on the MXK-194/198 (the MXK-194/198 is the DHCP server).

  Host-based routing with the MXK-194/198 as a local DHCP server on page 171, *Static and dynamic host configuration with the same subnet* on page 174 and *Host-based routing with the MXK-194/198 as a local DHCP server to provide DNS and bootp services* on page 175.

- The MXK-194/198 uses an external DHCP server.

  Host-based routing with an external DHCP server on page 179, *Host-based routing with multiple dhcp-relay agents and one DHCP server* on page 183, and *Host-based routing with an external DHCP server and an alternate DHCP server with dhcp-relay agent* on page 187.

For host based routing you first create a floating IP address (Numbered and unnumbered interfaces, floating interfaces, page 135), then associate the floating IP address with the physical interface. Each type of host-based router uses a different mechanism to associate the floating address with the physical interface:

- Static host-based interfaces

  The mechanism which associates the floating IP address and a static IP address given to an interface requires the static addresses must be in the same subnet as the floating address.

- DHCP server

  When the MXK-194/198 is a DHCP server, much like static addresses, the information in the **dhcp-server-subnet** which configures the network address of the subnet, the range of IP address given from the DHCP pool, and the default router must be in the same subnet as the floating address. The **dhcp-server-subnet** has an index which is then identified in the **host add dynamic** command to associate the physical interface with the DHCP server.

- DHCP relay agent

When the MXK-194/198 is a DHCP relay agent, an interface name is given to the floating IP address. In the **dhcp-relay add** command, the interface name is given, which associates the dhcp-relay agent with the floating IP address. The dhcp-relay agent creates a **dhcp-server-subnet** profile.The **host add dynamic** command uses the index from the **dhcp-server-subnet** to identify the physical interface with the DHCP relay agent.

Static host-based routing (without DHCP)

This procedure is for routing configurations statically without using DHCP, either locally or externally. Figure 41 displays a static host-based routing configuration.

Figure 41: Static host-based routing configuration

Configuring static host-based routing

To create static host-based routes, first you create the floating address, then using the **host add** commands configure the physical interface with the keyword **static** and the IP address. The static IP address given to the physical interfaces must be in the same subnet as the floating IP address.

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}  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}
```

3 Create GPON traffic profile.

```
zSH> new gpon-traffic-profile 1
gpon-traffic-profile  1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
```

```
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

4 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 1-1-6-1/gpononu gem 567 gtp 1 vlan 500 static 192.168.49.2
Adding host for 1-1-6-1/gpononu

zSH> host add 1-1-7-1/gpononu gem 534 gtp 1 vlan 600 static 192.168.49.3
Adding host for 1-1-7-1/gpononu

zSH> host add 1-1-8-1/gpononu gem 545 gtp 1 vlan 700 static 192.168.49.4
Adding host for 1-1-8-1/gpononu
```

5 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 1-1-6-567-gponport-500
Rd/Address         Interface                          Group T Host Address
-------------------------------------------------------------------------------
1 192.168.49.1     1-1-6-567-gponport-500             0     S 192.168.49.2
```

Verify the hosts (if necessary).

```
zSH> host show
Rd/Address         Interface                          Group T Host Address
-------------------------------------------------------------------------------
1 192.168.49.1     1-1-6-567-gponport-500             0     S 192.168.49.2
1 192.168.49.1     1-1-7-534-gponport-600             0     S 192.168.49.3
1 192.168.49.1     1-1-8-545-gponport-700             0     S 192.168.49.4
```

Deleting interfaces

1 Delete the GEM port. The static host IP interface on the GEM port will be deleted as well.

```
zSH> host delete 1-1-6-567/gponport vlan 500 all
Delete complete
```

2 Delete the floating IP interface.

```
zSH> interface delete float pmt1
Interface pmt1 deleted
```

Host-based routing with the MXK-194/198 as a local DHCP server

When configuring host-based routing with the MXK-194/198 as a local DHCP server, first create a floating IP interface, then create and configure a **dhcp-server-subnet** profile on the MXK-194/198. The **dhcp-server-subnet** profile is configured with the subnet IP address (**network** in the profile), the subnet mask and a range of addresses using the **range1-start** and **range1-end** to create the address pool. The subnet network address, range of addresses in the DHCP pool and default router address must be in the same subnet as the floating IP address. Multiple ranges may be configured.

The **dhcp-server-subnet** profile is associated with the floating IP interface to provide the IP address pool for the hosts. The subnet group number is assigned when creating the **dhcp-server-subnet** profile. The subnet group number (**subnetgroup**) is associated with the physical interface by the subnet group number in the **host add** command.

Figure 42 shows an MXK-194/198 as a local DHCP server.

Figure 42: MXK-194/198 as a local DHCP server

Configuring host-based routing with the MXK-194/198 as a local DHCP server

To create host-based routing with the MXK-194/198 as a local server you create the floating IP address.

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 pmt2 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 pmt2/ip
ip-interface-record pmt2/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  subnet network 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: s
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 1-1-7-1/gpononu gem 567 gtp 1 vlan 500 dynamic 1 5
Adding host for 1-1-7-1/gpononu

zSH> host add 1-1-8-1/gpononu gem 578 gtp 1 vlan 600 dynamic 1 5
Adding host for 1-1-8-1/gpononu
```

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 1-1-7-567-gponport-500
Rd/Address         Interface                          Group T Host Address
-------------------------------------------------------------------------------
1 10.107.8.254     1-1-7-567-gponport-500             1     D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>

zSH> host show 1-1-8-578-gponport-600
Rd/Address         Interface                          Group T Host Address
-------------------------------------------------------------------------------
1 10.107.8.254     1-1-8-578-gponport-600             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 GEM port. The static host IP interface on the GEM port will be deleted as well.* on page 170.

   host delete unused <*number*> deletes the designated number of unassigned floating IP slots.

```
zSH> host delete 1-1-8-578/gponport vlan 600 unused 5
Deleting host for 1-1-8-578/gponport
```

   host delete all deletes *all* of the host addresses on the designated interface, both assigned and unassigned.

```
zSH> host delete 1-1-7-567/gponport vlan 500 all
Deleting host for 1-1-7-567/gponport
```

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 float pmt2
Interface pmt2 deleted
```

Static and dynamic host configuration with the same subnet

The same subnet can be used for both static and dynamic configurations. Configuring dynamic hosts requires 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 Figure 43.

See *Host-based routing with the MXK-194/198 as a local DHCP server* on page 171 for configuring a **dhcp-server-subnet** profile.

Figure 43: Example dhcp-server-subnet profile for static and dynamic addresses using the same subnet

Host-based routing with the MXK-194/198 as a local DHCP server to provide DNS and bootp services

You can configure host-based routing with the MXK-194/198 as a local DHCP server to provide DNS and bootp services.

Configuring host-based routing with an MXK-194/198 as DHCP server to provide DNS and bootp services

With the MXK-194/198 as a DHCP server, additional services, such as DHCP and bootp services may be configured in the **dhcp-server-subnet** profile.

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 pmt3 10.107.8.1/24 255.255.255.0
Created ip-interface-record pmt3/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 pmt3/ip
ip-interface-record  pmt3/ip
1 entry found.
```

Verify the **ip-interface-record** profile for *pmt3*.

```
zSH> get ip-interface-record pmt3/ip
ip-interface-record pmt3/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}
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 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 44: DHCP server services available in the dhcp-server-subnet profile

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 1-1-7-1/gpononu gem 567 gtp 1 vlan 500 dynamic 3 2
Adding host for 1-1-7-1/gpononu
```

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 1-1-7-567-gponport-500
Rd/Address         Interface                          Group T Host Address
--------------------------------------------------------------------------------
1 10.107.8.1       1-1-7-567-gponport-500             3     D <unassigned>
                                                            D <unassigned>
```

Deleting the configuration

1 Delete the host.

```
zSH> host delete 1-1-7-567/gponport vlan 500 all
Deleting host for 1-1-7-567/gponport
```

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 pmt3
Interface pmt3 deleted
```

Host-based routing with an external DHCP server

Host-based routing on the MXK-194/198 with an external DHCP server, configures the MXK-194/198 to relay traffic between the hosts and the DHCP server. Figure 45 shows the MXK-194/198 as a DHCP relay agent with an external DHCP server.

Figure 45: MXK-194/198 as a DHCP relay agent with an external DHCP server

Note: You can configure the MXK-194/198 either as a local DHCP server or configure the MXK-194/198 to use an external DHCP server. The MXK-194/198 cannot be a local DHCP server and use an external DHCP on the same subnet.

However, you can use the MXK-194/198 as a local DHCP server and have an external DHCP if the subnets are not the same.

Configuring the MXK-194/198 for host-based routing with an external DHCP server

When creating a DHCP relay agent, the floating IP address is associated with the DHCP relay agent via an interface name in the **dhcp-relay add** command. The address of the remote DHCP server is also given in the **dhcp-relay add** command which creates a **dhcp-server-subnet profile** (with a subnetgroup index). The **host add dynamic** command associates the physical interface with the DHCP server via the subnet group index.

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 pmt4 192.168.49.1 255.255.255.0
Created ip-interface-record pmt4/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 pmt4/ip
ip-interface-record  pmt4/ip
1 entry found.
```

View the **ip-interface-record** profile for *pmt4/ip*.

```
zSH> get ip-interface-record pmt4/ip
ip-interface-record pmt4/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 pmt4
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*. It is important to the subnetgroup index in the **dhcp-server-subnet** as that index is used in the host add dynamic command.

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 1-1-5-1/gpononu gem 501 gtp 1 vlan 500 dynamic 1 3
Adding host for 1-1-5-1/gpononu
```

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 1-1-5-501-gponport-500
Rd/Address         Interface                          Group T Host Address
-------------------------------------------------------------------------------
1 10.107.8.254     1-1-5-501-gponport-500             1     D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>
```

Deleting the configuration

1 When necessary, delete the host.

```
zSH> host delete 1-1-5-501/gponport vlan 500 all
Deleting host for 1-1-5-501/gponport
```

2 Delete the dhcp-server subnet.

```
zSH> dhcp-relay delete 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 pmt4
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. Figure 46 shows an example of host-based routing with multiple subnets to one DHCP server.

Figure 46: Host-based routing with multiple subnets to one DHCP server

Configuring host-based routing with an external DHCP server and multiple dhcp-relay agents

Multiple subnets may be associated with a single external DHCP server by using the subnet group number. Each **dhcp-relay add** command will assign a new index which then is used in the **host add** command for each subnet.

1. Create more than one floating IP interface 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.
```

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.
```

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}systen 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 1-1-1-1/gpononu gem 501 gtp 1 vlan 500 dynamic 1 2
Adding host for 1-1-1-1/gpononu
```

Create the next host route designating the subnet group 2 and the number of floating IP addresses allowed.

```
zSH> host add 1-1-2-1/gpononu gem 601 gtp 1 vlan 600 dynamic 2 2
Adding host for 1-1-2-1/gpononu
```

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 1-1-1-501-gponport-500
Rd/Address         Interface                          Group T Host Address
-----------------------------------------------------------------------------
1 10.101.8.1       1-1-1-501-gponport-500             1     D <unassigned>
                                                            D <unassigned>

zSH> host show 1-1-2-601-gponport-600
```

```
Rd/Address         Interface                          Group T Host Address
-----------------------------------------------------------------------------
1 10.102.8.1       1-1-2-601-gponport-600             2     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 GEM port. The static host IP interface on the GEM port will be deleted as well.* on page 170.

   host delete unused <*number*> deletes the designated number of unassigned floating IP slots.

```
zSH> host delete 1-1-1-501/gponport vlan 500 unused 2
Deleting host for 1-1-1-501/gponport
```

   host delete all deletes *all* of the host addresses on the designated interface, both assigned and unassigned.

```
zSH> host delete 1-1-2-601/gponport all
Deleting host for 1-1-2-601/gponport
```

2. Delete the dhcp-server subnets.

```
zSH> dhcp-relay delete 1
Deleted DHCP Relay Agent number 1

zSH> dhcp-relay delete 2
Deleted DHCP Relay Agent number 2
```

3. Delete the floating interface.

```
zSH> interface delete float flt1
Interface flt1 deleted

zSH> interface delete float flt2
Interface flt2 deleted
```

Host-based routing with an external DHCP server and an alternate DHCP 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. Figure 47 shows an example of host-based routing with dhcp-relay and primary and alternate DHCP servers.

Figure 47: Host-based routing with dhcp-relay with a primary and alternate DHCP server

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 192.168.88.74 flt4
Created DHCP Relay Agent number 1
```

The DHCP relay agent is created with a DHCP server subnet group number of *1*.

3 Verify the **dhcp-server-subnet**.

```
zSH> get dhcp-server-subnet 1
dhcp-server-subnet  1
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: ----------->  {1}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 1-1-1-4/gpononu gem 501 gtp 1 vlan 500 dynamic 1 4
Adding host for 1-1-1-4/gpononu
```

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 1-1-1-501-gponport-500
Rd/Address         Interface                          Group T Host Address
-------------------------------------------------------------------------------
1 10.103.8.1       1-1-1-501-gponport-500             1     D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>
```

Deleting the configuration

1 When necessary, delete the host.

```
zSH> host delete 1-1-1-501/gponport vlan 500 all
```

```
Deleting host for 1-1-1-501/gponport
```

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 : yes
dhcp-server-subnet  1 deleted.
```

3 Delete the floating interfaces.

```
zSH> interface delete float flt4
Interface flt4 deleted
```

Host-based routing for data and voice services on GPON

This section describes the steps to create host-based routing for data and voice services on GPON.

To configure the MXK-194/198 for voice and data services create two different floating IP interfaces, one for each service.

Creating host-based routing for data and voice services on GPON

1 Create an IP interface on an Ethernet uplink port.

```
zSH> interface add 1-1-4-0/eth 192.169.1.14/24
Created ip-interface-record 1-1-4-0-eth/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
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 172.24.200.68/24    00:01:47:27:14:54       1-1-1-0-eth
1/1/4/0/ip     DOWN    1 192.169.1.14/24     00:01:47:27:14:55       1-1-4-0-eth
-------------------------------------------------------------------------------
```

2 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 Verify the floating IP interfaces.

```
zSH> list ip-interface-record flt1/ip
ip-interface-record  flt1/ip
1 entry found.

zSH> list ip-interface-record flt2/ip
ip-interface-record  flt2/ip
1 entry found.
```

3 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 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 Verify the **dhcp-server-subnet**(s).

```
zSH> list dhcp-server-subnet
dhcp-server-subnet  1
dhcp-server-subnet  2
2 entries found.
```

4 Create the host routes for voice and data 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 1-1-1-5/gpononu gem 501 gtp 2 vlan 400 dynamic 1 5
Adding host for 1-1-1-5/gpononu
```

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 1-1-2-6/gpononu gem 601 gtp 2 vlan 600 dynamic 2 1
Adding host for 1-1-2-6/gpononu
```

c Verify the hosts.

```
zSH> host show 1-1-1-501-gponport-400
Rd/Address         Interface                          Group T Host Address

-----------------------------------------------------------------------------------------------------------
1 192.168.49.1     1-1-1-501-gponport-400             1     D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>
                                                            D <unassigned>
zSH> host show 1-1-2-601-gponport-600
Rd/Address         Interface                          Group T Host Address
-----------------------------------------------------------------------------------
1 192.168.48.1     1-1-2-601-gponport-600             2     D <unassigned>
```

Deleting the data and voice configuration

Delete the data and voice configuration.

1 Delete the host routes.

```
zSH> host delete 1-1-1-501/gponport vlan 400 all
Deleting host for 1-1-1-501/gponport

zSH> host delete 1-1-2-601/gponport vlan 600 all
Deleting host for 1-1-2-601/gponport
```

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 : yes
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 : yes
dhcp-server-subnet  2 deleted.
```

3 Delete the floating IP interfaces.

```
zSH> delete ip-interface-record  flt1/ip
ip-interface-record  flt1/ip
1 entry found.
Delete ip-interface-record flt1/ip? [y]es, [n]o, [q]uit : yes
ip-interface-record  flt1/ip deleted.

zSH> delete ip-interface-record  flt2/ip
ip-interface-record  flt2/ip
1 entry found.
Delete ip-interface-record flt2/ip? [y]es, [n]o, [q]uit : yes
ip-interface-record  flt2/ip deleted.
```

IP administrative procedures

The following IP administrative procedures are supported on the MXK-194/198:

- Modify profiles created by host/interface add commands, page 192
- Display hosts, page 192
- Display interfaces, page 194
- Display routing information, page 194
- Delete hosts, page 195
- Delete routes, page 196
- DHCP logging, page 196
- IP statistics commands, page 198

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-194/198. 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     1-1-2-456-gponport-400             1     D <unassigned>
```

```
                                                                D <unassigned>
                                                                D <unassigned>
                                                                D <unassigned>
                                                                D <unassigned>
1 10.107.8.254     1-1-3-457-gponport-600              1      D <unassigned>
                                                                D <unassigned>
```

Filtering host show

An unfiltered **host show** command, displaying all of the host interfaces for an MXK-194/198 system may create a long list which makes it hard to find the actual interfaces for which you are looking.

Unfiltered host show

To highlight the different **host show** filtering examples, this first **host show** example displays a variety of host interfaces

```
zSH> host show
Rd/Address         Interface                           Group T Host Address
--------------------------------------------------------------------------------
1 10.107.8.254     1-1-2-456-gponport-400              1      D <unassigned>
                                                                D <unassigned>
                                                                D <unassigned>
                                                                D <unassigned>
                                                                D <unassigned>
1 10.107.8.254     1-1-3-457-gponport-600              1      D <unassigned>
                                                                D <unassigned>
```

Filtering host show by VLAN

```
zSH> host show vlan 400
Rd/Address         Interface                           Group T Host Address
--------------------------------------------------------------------------------
1 10.107.8.254     1-1-2-456-gponport-400              1      D <unassigned>
                                                                D <unassigned>
                                                                D <unassigned>
                                                                D <unassigned>
                                                                D <unassigned>
```

Filtering host show by port

```
zSH> host show 1-1-2-456-gponport-400
Rd/Address         Interface                           Group T Host Address
--------------------------------------------------------------------------------
1 10.107.8.254     1-1-2-456-gponport-400              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
4 interfaces
Interface      Status  Rd/Address            Media/Dest Address      IfName
-------------------------------------------------------------------------------
1/1/1/0/ip     UP      1 10.55.1.198/24      00:01:47:24:08:ae       1-1-1-0-eth
1/1/2/456/ip   DOWN    1 [10.107.8.254]                      1-1-2-456-gponport-400
1/1/3/457/ip   DOWN    1 [10.107.8.254]                      1-1-3-457-gponport-600
1/1/4/0/ip     DOWN    1 192.169.1.14/24     00:01:47:24:08:b0       1-1-4-0-eth-777
-------------------------------------------------------------------------------
```

Display routing information

This section discusses:

- Displaying the routing table, page 194
- Displaying RIP information, page 194

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                10.55.1.254      1        STATICLOW
10.55.1.0/24             1/1/1/0/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
-------------------------------------------------------
                    Recv Bad    Recv Bad    Updates
```

```
IfName              Packets     Routes      Sent To
-----------------------------------------------------
1-1-1-0-eth         0           0           0
1-1-4-0-eth-777     0           0           0
1-1-2-456-gponport  0           0           0
1-1-3-457-gponport  0           0           0
RIP Interface Configuration
-----------------------------------------------------------------------------------
-------------------
                    Auth    Auth                              Default  Src
IfName              Type    Key          Talk      Listen    Metric   Address
Static  Poison
-----------------------------------------------------------------------------------
-------------------
1-1-1-0-eth         none    (write-only)  disabled  disabled  0       10.55.1.198
none    disabled
1-1-4-0-eth-777     none    (write-only)  disabled  disabled  0       192.169.1.14
none    disabled
1-1-2-456-gponport  none    (write-only)  disabled  disabled  0       0.0.0.0
none    disabled
1-1-3-457-gponport  none    (write-only)  disabled  disabled  0       0.0.0.0
none    disabled
RIP Peers
-------------------------------------------------------------------------------
Route       IP               Last                          Recv Bad    Recv Bad
Domain      Address          Update               Version  Packets     Routes
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
```

Delete hosts

There are several ways to use **host delete** to delete IP interfaces associated with an *interface/type*.

Deleting hosts using unused

host delete *interface/type unused* <*number*> deletes the designated number of unassigned floating IP slots that have not yet been assigned an IP address.

```
zSH> host delete 1-1-1-501/gponport vlan 500 unused 2
Deleting host for 1-1-1-501/gponport
```

Deleting hosts using all

host delete all deletes *all* of the hosts on this subnet and the subnet itself.

```
zSH> host delete 1-1-1-501/gponport vlan 500 all
Deleting host for 1-1-1-501/gponport
```

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, page 196
- DHCP server log messages, page 197
- View client leases, page 197

Enable DHCP logging

The MXK-194/198 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-194/198 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-194/198, 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: ------------->  {hdsl2}
adminstatus: ------>  {up}
physical-flag: ---->  {true}
iftype-extension: ->  {none}
ifName: ----------->  {1-10-48-0}
```

The MXK-194/198 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-194/198 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:
        8 calls to icmp_error
        0 error not generated because old message was icmp
        Output histogram:
                echo reply: 1
                destination unreachable: 8
        0 message with bad code fields
        0 message < minimum length
        0 bad checksum
        0 message with bad length
        Input histogram:
                echo: 1
        1 message response generated
```

- **ip ifstat**

  Displays interface statistics

```
zSH> ip ifstat
```

```
Interface                    Received Packets        Transmitted Packets
Name                        UCast   MCast   BCast    UCast   MCast   Bcast
lo                          9019    0       0        9019    0       0
1-1-1-0-eth                 9079    106019  0        9109    2       0
2 interfaces
```

- **ip ifsum**

  Displays a summarized list of known interfaces.

```
zSH> ip ifsum
lo SOFTWARELOOPBACK ifindex 1 (ifp 0x1ba6088, 5|2)
     Flags: UP LOOPBACK MCAST ARP RUNNING
     inet 127.0.0.1 netmask 255.0.0.0
1-1-1-0-eth ETHERNETCSMACD ifindex 63 (ifp 0x5565d58, 9|4)
     Flags: UP BCAST MCAST IPFWD MCASTFWD ARP RUNNING CFGCURRENT
     inet 172.24.200.68 netmask 255.255.255.0  bcast 172.24.200.255
2 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)
-------- ----- ------ ------  ------------------       ------------------       -------
7cd6fc8  TCP        0    242  172.24.200.68.23         172.16.48.178.1292       ESTABLISHED
7cd6aa0  TCP        0      0  0.0.0.0.22               0.0.0.0.0                LISTEN
7cd6890  TCP        0      0  0.0.0.0.23               0.0.0.0.0                LISTEN
7cd6ec0  UDP        0      0  0.0.0.0.67               0.0.0.0.0
7cd6e3c  UDP        0      0  0.0.0.0.68               0.0.0.0.0
7cd6db8  UDP        0      0  0.0.0.0.69               0.0.0.0.0
7cd6d34  UDP        0      0  0.0.0.0.520              0.0.0.0.0
7cd6ba8  UDP        0      0  0.0.0.0.162              0.0.0.0.0
7cd6b24  UDP        0      0  0.0.0.0.161              0.0.0.0.0
7cd6a1c  UDP        0      0  0.0.0.0.0                0.0.0.0.0
7cd6998  UDP        0      0  127.0.0.1.1025           127.0.0.1.1024
7cd6914  UDP        0      0  0.0.0.0.1024             0.0.0.0.0
7cd6f44  RAW        0      0  0.0.0.0.0                0.0.0.0.0
7cd6cb0  RAW        0      0  0.0.0.0.0                0.0.0.0.0
7cd6c2c  RAW        0      0  0.0.0.0.0                0.0.0.0.0
```

- **ip ipstat**

  Displays IP statistics.

```
zSH> ip ipstat
                total 12837
               badsum    0
             tooshort    0
             toosmall    0
              badhlen    0
               badlen    0
          infragments    0
          fragdropped    0
          fragtimeout    0
```

```
              forward 9036
          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 12705
  pendingarpoverflow    5
```

- **ip tcpstat**

  Displays TCP statistics.

```
zSH> ip tcpstat
TCP:
        9071 packets sent
                5501 data packets (54891 bytes)
                0 data packet (0 byte) retransmitted
                3570 ack-only packets (2 delayed)
                0 URG only packet
                0 window probe packet
                0 window update packet
                0 control packet
        9057 packets received
                5470 acks (for 54890 bytes)
                18 duplicate acks
                0 ack for unsent data
                4895 packets (6171 bytes) received in-sequence
                0 completely duplicate packet (0 byte)
                0 packet with some dup. data (0 byte duped)
                0 out-of-order packet (0 byte)
                0 packet (0 byte) of data after window
                0 window probe
                0 window update packet
                0 packet received after close
```

```
                0 discarded for bad checksum
                0 discarded for bad header offset field
                0 discarded because packet too short
        0 connection request
        1 connection accept
        1 connection established (including accepts)
        0 connection closed (including 0 drop)
        0 embryonic connection dropped
        5469 segments updated rtt (of 5470 attempts)
        0 retransmit timeout
                0 connection dropped by rexmit timeout
        0 persist timeout
        18 keepalive timeouts
                18 keepalive probes sent
                0 connection dropped by keepalive
        0 pcb cache lookup failed
        0 mama cache lookup failed
        0 mama flings
        0 mama alloc drops
```

- **ip udpstat**

  Displays UDP statistics.

```
zSH> ip udpstat
UDP:
        3916 total packets
        3791 input packets
        125 output packets
        0 incomplete header
        0 bad data length field
        0 bad checksum
        3654 broadcasts received with no ports
        0 full socket
        0 allocated but not bound drops
        125 pcb cache lookups failed
        0 pcb hash lookup failed
        0 mama cache lookup failed
        0 packets flung to other card
```

- **ip arpshow**

  Displays the ARP table.

```
zSH> ip arpshow
LINK LEVEL ARP TABLE
destination      gateway              flags     Refcnt  Use           Interface
-------------------------------------------------------------------------------
172.24.200.68    00:01:47:27:14:54     00405    1       646           lo
172.24.200.252   00:04:4d:47:bd:c2     00405    0       0             1-1-1-0-eth
172.24.200.254   00:00:0c:07:ac:0      00405    1       0             1-1-1-0-eth
-------------------------------------------------------------------------------
        6 routes, 3 arp routes
-------------------------------------------------------------------------------
```

- **ip arpdelete** *ipaddress*

  Deletes an entry from the ARP table.

- **ip arpflus**h

  Flushes the ARP table of all entries.

8 BRIDGING CONFIGURATION

This chapter covers Zhone's bridging services including:

- Overview, page 204
- Terminology and concepts, page 205
- SLMS bridge types, page 213
- Tagging operations, page 229
- Common bridge commands, page 235
- Settings for asymmetric bridges, page 236
- Settings for symmetric bridges, page 237
- Shaping Traffic: Class of Service Queuing, page 239
- Filters for MXK-194/198 (packet-rule-record), page 241
- Advanced bridging topics, page 267
- VLAN translation, page 281
- MXK-194/198 bridging configurations, page 281
- Administrative commands, page 304

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 MXK-194/198 and CPEs to form a myriad of Zhone solutions which supports a flexible network design.

The MXK and MALC platforms blend high performance and high density integrated access platforms allowing providers the flexibility to match their network to the needs of their markets. The MXK-194/198 can 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 completes a very effective access picture.

Bridging services are usually configured with 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 the Ethernet, or GPON interface.

This chapter describes the basic concepts for understanding how to build bridges using the MXK-194/198 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 (OSI Logical Link layer 2, bridging)
- Packets are delivered based on IP address (OSI 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: OSI Open Systems Interconnection Reference Mode l

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 Figure 1. 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, page 206
- Physical interface, page 206
- Logical interface, page 207
- Bridges and bridge interfaces, page 207
- VLANs and SLANs, untagged, tagged and stagged, page 207
- Upstream and downstream, page 210
- Broadcast, multicast, and unicast, page 211

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
- GPON ONU 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 Figure 2.

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 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, a GPON Virtual Connection (GEM port) for connection based technologies such as GPON, or both.

For information about IP interfaces, see *IP Configuration* on page 131.

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 page 213.

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:

- providing port-to-port security of users sharing a VLAN as with Destination MAC swapping.

  For details see *Destination MAC swapping (dstmacswapdynamic and dstmacswapstatic)* on page 265

- inserting identification information for DHCP servers

  For details see *DHCP on bridge packet rules (DHCP relay, and Forbid OUI)* on page 260

- inserting tags for identification purposes as when the MXK-194/198 is a PPPoE intermediate agent

  For details see *PPPoE with intermediate agent (bridgeinsertpppoevendortag)* on page 245

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 Figure 3.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 *stagged* as shown in Figure 4.

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 stagged frames. No other frames will be accepted. If a bridge interface is expecting a tagged frame (using the **bridge add** command with the *tagged* keyword 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* keyword or not using the keywords to define single tagged (*tagged)* or double tagged (*stagged)*.

You can issue a **bridge add** command without specifying whether the bridge interface is to be *untagged*, *tagged* or *stagged*. 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

  untagged. In this case, must designate a VLAN or SLAN.

See *Tagging operations* on page 229 for more information on untagged, tagged, and stagged 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 Figure 5.

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 Figure 4 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.

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, MX, MXP, 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-194/198 system.

Asymmetric bridges use three different bridge interface types:

- Uplink

  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.

  Uplink bridge interfaces only work in conjunction with asymmetric bridge interfaces.

- Downlink

  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.

- Intralink

  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 6: TLS bridges joining remote segments 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 Figure 8.

Figure 8: With TLS bridges all interfaces learn on ingress

Configure a TLS bridge

For TLS bridges only, the first instance of a TLS bridge with VLAN ID, regardless of network facing or subscriber facing, associates a bridge-path with the configured VLAN ID.

The configurable parameters for the bridge-path that are relevant to TLS bridges are the aging period with a default of *3600*, and the flap control with a default of *fast*.

The default of *fast* indicates that as a MAC address comes into the system from one source and then is seen from another source, the MAC address table is purged from the first source and replaced with the second source without delay or restriction. If this behavior is not desired, the Flap Mode can be configured to *disabled* or *default*.

The default age of *3600* is how long a MAC address is held in the MAC table before it is purged. This time is now configurable on TLS bridges.

The MCAST and IGMP Query Interval are not relevant to TLS bridges.

Configuring a TLS bridge

For example, the first TLS bridge on a subscriber facing port for VLAN ID 100:

1 For each connection to the TLS bridge add, a *tls* bridge interface

```
zSH> bridge add 1-1-2-0/eth tls vlan 100
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record 1-1-2-0-eth/bridge
Bridge-path added successfully
```

TLS bridges can be thought of as a community since they share traffic much in the way a physical LAN shares traffic.

2 Create a GPON traffic profile.

```
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}:
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 Specify the GTP ID when creating GEM port with the **bridge add** command. For each connection to the TLS bridge, add a *tls* bridge interface:

```
zSH> bridge add 1-1-4-5/gpononu gem 501 gtp 1 tls vlan 100
Adding bridge on 1-1-4-5/gpononu
Created bridge-interface-record 1-1-4-501-gponport-100/bridge
Bridge-path added successfully
```

This example uses a Dynamic OMCI provisioned ONT, if the ONT is provisioned by Smart OMCI, the **bridge add** command will be different. Check the *Chapter 11, GPON Subscriber Interfaces* for the details.

The TLS bridge interfaces with VLAN 100 will all work together as one TLS bridge.

4 Verify the bridges.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------------
tls               100       1/1/2/0/eth        1-1-2-0-eth/bridge                  DWN
tls               100       1/1/4/1/gpononu    1-1-4-501-gponport-100/bridge       DWN
2 Bridge Interfaces displayed
```

5 Verify the bridge-path.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                                Address
-------------------------------------------------------------------------------
       100 N/A                                VLAN, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Fast
```

Note: When you do not specify *untagged*, *tagged*, or *stagged* to the bridge interface, the interface will use the default for TLS bridges, which is *untagged*.

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. MXK-194/198 may have multiple intralinks per bridge, but other SLMS devices may only have one intralink. There may be multiple downlinks.

Typically 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. Intralinks have different learning behavior than uplinks or downlinks.

When setting up Internet access for multiple subscribers you configure the MXK-194/198 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 Figure 10.

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 Figure 11.

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.

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 Dynamic Host Configuration Protocol (DHCP) and Address Resolution Protocol (ARP) exceptions noted in the next section.

Custom DHCP

DHCP servers provide a pool of IP addresses, and upon request provide an IP address for a device. When a MXK-194/198 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.

The customDHCP (bridgeIfCustomDHCP) setting can be found in the **bridge-interface-record** that created by the **bridge add** command. The **list bridge-interface-record** command will list all bridge-interface-records. The **update bridge-interface-record** command may be used to modify the customDHCP setting. The **get bridge-interface-record** command may be used to view the current bridge settings.

Custom ARP

Broadcast frames received on the uplink bridge interface in an asymmetric bridge are blocked. ARP and 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.

The customARP setting can be found in the **bridge-interface-record** created from the **bridge add** command. The **update bridge-interface-record** command may be used to modify the customARP setting.

How ARP frames are handled is dependent on the **customARP** parameter in the **bridge-interface-record** 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 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 not a match, then the ARP is filtered and the MXK-194/198 will flood the broadcast out all other bridge interfaces

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 and a bridge path to send frames received on the downlinks. (See *Bridge add and bridge-path add defaults* on page 222 for a discussion on automatically created bridge paths and bridge-path parameters). You must also designate a VLAN ID to match any VLAN IDs configured on the downlinks.

Configuring an uplink bridge

1. Add a bridge interface on the uplink interface.

```
zSH> bridge add 1-1-3-0/eth uplink vlan 500 tagged
Adding bridge on 1-1-3-0/eth
Created bridge-interface-record 1-1-3-0-eth-500/bridge
Bridge-path added successfully
```

2. Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
```

```
compensated: ------------>  {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.
```

3 Add downlink bridge interfaces.

```
zSH> bridge add 1-1-1-3/gpononu gem 501 gtp 1 downlink vlan 500
Adding bridge on 1-1-1-3/gpononu
Created bridge-interface-record 1-1-3-501-gponport/bridge
```

4 Verify the bridges.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------------
upl            Tagged  500    1/1/3/0/eth             1-1-3-0-eth-500/bridge        DWN S VLAN 500 default
dwn                500        1/1/3/1/gpononu         1-1-3-501-gponport/bridge          DWN
2 Bridge Interfaces displayed
```

Bridge add and bridge-path add defaults

The system automatically creates a bridge-path with default values when entering the **bridge add** command for uplink, TLS, and intralink bridges.

Default bridge-path for uplink bridge:

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
      999 1-1-2-0-eth-999/bridge            Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default

zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
       444 1-1-4-0-eth-444/bridge             Intralink

zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
       800 N/A                                VLAN, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Fast
```

There are optional arguments for the bridge that must be configured from the CLI with the **bridge-path add** command. These include:

- age
- multicast aging period (mcast)
- flap control (flap)
- unicast aging period (age)
- IGMP query interval (igmpsnooping)
- IGMP timer
- flags

When the **bridge-path add** command is entered for an existing bridge, the previously existing bridge path is overwritten and unless otherwise specified, any previously existing optional arguments will revert to their default.

In other words, if the existing bridge path includes a designation for flap control and you want to add IGMP timer, you must enter both the flap control value *and* the IGMP timer value. Otherwise the flap control value will revert to the default.

For example, parameters such as *mcast* and *igmp* for video bridging, enter the **bridge-path add** command with the proper variables.

The following example show a bridge added and the bridge-path automatically created.

```
zSH> bridge add 1-1-2-0/eth uplink vlan 999
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record 1-1-2-0-eth-999/bridge
Bridge-path added successfully

zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------------------
upl           Tagged  999    1/1/2/0/eth             1-1-2-0-eth-999/bridge        DWN S VLAN 999 default
1 Bridge Interfaces displayed

zSH> bridge-path show
VLAN/SLAN   Bridge                                Address
-------------------------------------------------------------------------------
      999 1-1-2-0-eth-999/bridge            Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
```

The following example shows the **bridge-path add** command used to add parameters to the bridge.

```
zSH> bridge-path add 1-1-2-0-eth-999/bridge vlan 999 default igmptimer 30 igmpsnooping
enable
Bridge-path added successfully

zSH> bridge-path show
VLAN/SLAN   Bridge                                Address
-------------------------------------------------------------------------------
```

```
      999 1-1-2-0-eth-999/bridge            Default, Age: 3600, MCAST Age: 61, IGMP
Query Interval: 30, IGMP Proxy, Flap Mode: Default
```

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-194/198. If a frame is received on a downlink, the MAC address is learned. When the frame is received on the uplink with a known address, the frame is forwarded to the downlink with that address. When 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-194/198 in the chain or the upper MXK-194/198s 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.

Figure 12 shows downlinks to GPON zNIDs and intralinks from an MXK to subtended MXK-194/198s. 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.

Figure 13: The intralink portion of an asymmetric bridge

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 MXK-194/198s

This example adds an intralink bridge interface to an asymmetric uplink/downlink bridge.

Configuring MXK-194/198 intralinks

Configure the MXK-194/198 for intralinked bridges.

1. Add bridge interfaces on the uplink interface. For each VLAN ID designated on a downlink, there must be an uplink with the corresponding VLAN ID.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 100
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-100/bridge
Bridge-path added successfully
zSH> bridge add 1-1-4-0/eth uplink vlan 200
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-200/bridge
Bridge-path added successfully
```

The bridge path automatically created designates the bridge as default. Default designates that all unknown unicast frames will be sent to the intralink rather than discarded as with an asymmetric bridge without an intralink.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
--------------------------------------------------------------------------------
      100 1-1-4-0-eth-100/bridge            Default, Age: 3600, MCAST Age: 150, IGMP Query
Interval: 70, Flap Mode: Default
      200 1-1-4-0-eth-200/bridge            Default, Age: 3600, MCAST Age: 150, IGMP Query
Interval: 70, Flap Mode: Default
```

Note: The MXK-194/198 does not support the keyword *global*. For each VLAN or SLAN, the uplink bridge must be set to default.

2 Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

3 Add downlink bridges for devices downstream from the MXK-194/198.

```
zSH> bridge add 1-1-1-4/gpononu gem 501 gtp 1 downlink vlan 100
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-501-gponport-100/bridge

zSH> bridge add 1-1-1-3/gpononu gem 501 gtp 1 downlink vlan 200
Adding bridge on 1-1-1-3/gpononu
```

```
Created bridge-interface-record 1-1-3-501-gponport-200/bridge
```

4 Create a bridge interface for the intralink with a VLAN ID.

The intralink can be between the MXK319 and a subtended MXK-194/198, MALC, or SLMS device. Then add the bridge path for the intralink.

```
zSH> bridge add 1-1-2-0/eth intralink vlan 444
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record 1-1-2-0-eth-444/bridge
Bridge-path added successfully

zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
      100 1-1-4-0-eth-100/bridge            Default, Age: 3600, MCAST Age: 150, IGMP Query
Interval: 70, Flap Mode: Default
      200 1-1-4-0-eth-200/bridge            Default, Age: 3600, MCAST Age: 150, IGMP Query
Interval: 70, Flap Mode: Default
       444 1-1-2-0-eth-444/bridge               Intralink
```

This command mainly defines the behavior that source addresses from the intralink will not be learned.

Note: The MXK-194/198 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 1-1-3-0/eth uplink vlan 444
Adding bridge on 1-1-3-0/eth
Created bridge-interface-record 1-1-3-0-eth-444/bridge
Bridge-path added successfully
```

6 Verify the bridges created.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                       St  Table Data
-----------------------------------------------------------------------------------------
int            Tagged  444    1/1/2/0/eth             1-1-2-0-eth-444/bridge       UP  S VLAN 444 Intralink
upl            Tagged  444    1/1/3/0/eth             1-1-3-0-eth-444/bridge       UP  S VLAN 444 default
upl            Tagged  100    1/1/4/0/eth             1-1-4-0-eth-100/bridge       DWN S VLAN 100 default
upl            Tagged  200    1/1/4/0/eth             1-1-4-0-eth-200/bridge       DWN S VLAN 200 default
dwn                100        1/1/4/1/gpononu         1-1-4-501-gponport-100/bridge DWN
dwn                200        1/1/3/1/gpononu         1-1-3-501-gponport-200/bridge DWN
6 Bridge Interfaces displayed
```

7 Verify the bridge paths.

```
zSH> bridge-path show
```

```
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
      100 1-1-4-0-eth-100/bridge            Default, Age: 3600, MCAST Age: 150, IGMP Query
Interval: 70, Flap Mode: Default
      200 1-1-4-0-eth-200/bridge            Default, Age: 3600, MCAST Age: 150, IGMP Query
Interval: 70, Flap Mode: Default
       444 1-1-2-0-eth-444/bridge             Intralink
      444 1-1-3-0-eth-444/bridge            Default, Age: 3600, MCAST Age: 150, IGMP Query
Interval: 70, Flap Mode: Default
```

Tagging operations

This section describes VLAN and SLAN tagging operations including:

- Overview, page 229
- Common tagging operation scenarios, page 231
- Tagging operations, page 229

Overview

VLANs and SLANs (see *VLANs and SLANs, untagged, tagged and stagged* on page 207 for information about VLANs and SLANs) define the bridge to which an incoming frame belongs. The bridge type — as discussed in *SLMS bridge types* on page 213 — 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 stagged 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 Figure 14, The MXK-194/198s 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 MXK-194/198.

In the example from Figure 14, 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 MXK-194/198 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 Figure 14, 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.

Zhone does not support stagged with known VLAN ID and unknown SLAN ID.

Note: The MXK-194/198 does not support stagged frames with unknown VLAN and unknown SLAN.

The frames which come into the MXK-194/198 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 Figure 15, the MXK-194/198 is the next level up from the GPON zNID and acts as line concentrator. The example shows only VLAN tagging.

Figure 15: GPON zNID providing edge tagging, MXK-194/198 as line concentrator

Figure 15 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.

The uplinks can be separated by VLAN which is a common scenario, as shown in Figure 16. Normally in a triple play scenario you would have separate VLANs for video, voice, or data services. In this way you can keep known traffic separate for defining QoS prioritization or other bridge additions as provided by packet rules or GPON Traffic Profiles (GTPs).

Figure 16: OMCI GPON GEM port encapsulation to separate private VLANs

A good way to learn tagging fundamentals is by exploring some of the common scenarios. Figure 14 shows promoting (and stripping) VLAN tags at the network edge. Figure 15 shows that same promotion at the edge, but now a line concentrator (in the example a MXK-194/198) distributes access from many downstream lines to a trunk. These multiple downstream subscriber lines could be configured for different services, such as data, voice, and video, but all in GPON transport technologies (see Figure 16).

Figure 17: Parts of the bridge add command in Dynamic OMCI

Figure 18: Parts of the bridge add command in Smart OMCI

GEM port 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 GPON specific portion for GPON transport media.

As shown in Figure 17 and Figure 18, the **bridge add** command has different formats for Dynamic OMCI and Smart OMCI. For details, refer to *Chapter 11, GPON Subscriber Interfaces,* on page 329.

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.

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.

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.

The flexibility of the SLMS tagging mechanism works for many scenarios. Not only do the MXK and MXK-194/198 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 19: SLMS devices support untagged on upstream interface

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.

Common bridge commands

This section describes:

- bridge add command, page 235
- Verifying bridge interface settings, page 235

Note that the **bridge add** command examples used in this chapter are in Smart OMCI format.

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 *Bridge add and bridge-path add defaults* on page 222 for information on how to use the **bridge-path add** command).

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 page 267 for options.

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 1-1-1-5-gponport-840/bridge
bridge-interface-record  1-1-1-509-gponport-840/bridge
vpi: --------------------------------->  {0}
vci: --------------------------------->  {0}
vlanId: ------------------------------->  {840}
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}
vlan-xlate-from: --------------------->  {0}
slan-xlate-from: --------------------->  {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.

Settings for asymmetric bridges

Table 2 lists the default asymmetric **bridge-interface-record** settings for the supported bridge options.

Table 2: Default values for asymmetric bridge-interface-record

Parameter	Uplink	Downlink Untagged	Downlink Tagged	Intralink Tagged
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

Table 2: Default values for asymmetric bridge-interface-record (Continued)

Parameter	Uplink	Downlink Untagged	Downlink Tagged	Intralink Tagged
bridgeIfIngressPacketRule GroupIndex	0	0	0	0
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
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
bridgeIfEgressPacketRule GroupIndex:	0	0	0	0
bridgeIfTableBasedFilter	NONE(0)	NONE(0)	NONE(0)	NONE(0)
bridgeIfDhcpLearn	NONE(0)	NONE(0)	NONE(0)	NONE(0)

Settings for symmetric bridges

Table 3 lists the default **bridge-interface-record** settings for the supported symmetric bridge options.

Table 3: Default values for TLS bridge-interface-record

Parameter	TLS
vlanId	As specified
stripAndInsert	True
customARP	False
filterBroadcast	False

Table 3: Default values for TLS bridge-interface-record (Continued)

Parameter	TLS
learnIP	False
learnUnicast	True
maxUnicast	100
learnMulticast	False
forwardToUnicast	True
forwardToMulticast	False
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
```

```
      Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
--------------------------------------------------------------------------------------------
int             Tagged  444    1/1/2/0/eth              1-1-2-0-eth-444/bridge        UP  S VLAN 444 Intralink
tls                 300        1/1/5/0/eth              1-1-5-0-eth/bridge           DWN
upl             Tagged  444    1/1/3/0/eth              1-1-3-0-eth-444/bridge        UP  S VLAN 444 default
upl             Tagged  100    1/1/4/0/eth              1-1-4-0-eth-100/bridge        DWN S VLAN 100 default
upl             Tagged  200    1/1/4/0/eth              1-1-4-0-eth-200/bridge        DWN S VLAN 200 default
dwn                 100        1/1/4/1/gpononu          1-1-4-501-gponport-100/bridge DWN
dwn                 200        1/1/3/1/gpononu          1-1-3-601-gponport-200/bridge DWN
tls                 300        1/1/5/1/gpononu          1-1-5-701-gponport-300/bridge DWN
8 Bridge Interfaces displayed
```

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-194/198 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 out a uplink, intralink, or downlinked tagged bridge. The configured CoS level specifies the packet priority and queueing methods used to transport the packet through the Ethernet network. The MXK-194/198 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.

Traffic is prioritized across MXK-194/198 based on the VLAN CoS of the incoming frames. However, the MXK-194/198 supports four queues per physical interface, so frames with a 0 or 1 CoS value are put into queue number 1; frames with a 2 or 3 CoS value are put into queue number 2; frames with a 4 or 5 in queue number three; and 6 or 7 in queue number 4.

These are strict priority queues which mean that everything is cleared out of the high priority queue first (queue number 4 with CoS values 6 or 7) Only after that queue is empty is the next queue (number 3) 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.

Configuring Class of Service

The following parameters in the bridge interface record are used for Ethernet COS support.

Table 4: 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 - 4095}
vci:---------------------------------->   {0 - 65535}
vlanId:------------------------------->   {0 - 4094}
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}
```

Adding a bridge with a CoS value

This example adds GEM port 501 on GPON ONU interface *1-1-1-4* 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. Make sure the GTP exists before specifying its index in the **bridge add** command.

```
zSH> bridge add 1-1-1-4/gpononu gem 501 gtp 1 downlink vlan 100 tagged COS 7
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-501-gpononu-100/bridge
```

This example adds GEM port 501 on GPON ONU interface *1-1-1-4* with a vlanIDCOS value of 7 and enables the overwriting of the VLAN ID in all outgoing packets with the value of 7.

```
zSH> bridge add 1-1-1-5/gpononu gem 501 gtp 1 downlink vlan 100 tagged cos 7
outcosall 7
Adding bridge on 1-1-1-5/gpononu
Created bridge-interface-record 1-1-5-501-gpononu-100/bridge
```

Filters for MXK-194/198 (packet-rule-record)

This section explains how to configure **packet-rule-record** filters and includes:

- Overview of packet-rule-record filters, page 241
- Configure packet-rule-records, page 244
- PPPoE with intermediate agent (bridgeinsertpppoevendortag), page 245
- Option 82 DHCP on bridge packet rule (bridgeinsertoption82), page 253
- DHCP on bridge packet rules (DHCP relay, and Forbid OUI), page 260
- Destination MAC swapping (dstmacswapdynamic and dstmacswapstatic), page 265

Related section:

Bridge configuration with DHCP relay, page 260

Overview of packet-rule-record filters

The SLMS CLI architecture has a mechanism for adding multiple filters for ingress interfaces by grouping **packet-rule-record**(s).

Configure the packet rule group index in the **bridge-interface-record**. Multiple bridges may use the same interface packet rule group index as shown in Figure 20.

Figure 20: Multiple filters for bridge interfaces

Create packet-rule-record filters

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"
Created packet-rule-record 10/2 (bridgeinsertoption82)
```

```
zSH> rule add bridgeinsertoption82 10/3 "circuitIDExample" "remoteIDExample"
Created packet-rule-record 10/3 (bridgeinsertoption82)
```

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.

Table 5 lists which packet rule types function on ingress (**ipktrule**) and which ones function on egress (**epktrule**).

Table 5: Bridge packet rules function on ingress or egress

Packet Rules	Ingress or Egress
bridgeinsertoption82	Ingress
bridgeinsertpppoevendortag	Ingress
bridgedhcprelay	Ingress
bridgeforbidoui	Ingress
dscptocos	Ingress **Note:** dscptocos bridge packet rule is not supported on the MXK-194/198.
ratelimitdiscard	Ingress, Egress
colorawareratelimitdiscard	Ingress, Egress
dstmacswapstatic	Egress (uplink)
dstmacswapdynamic	Egress (uplink)
filterfirstencapsulationvlan	Ingress
promotefirstencapsulationvlan	Ingress
deny	Ingress, 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 1-1-1-4/gpononu gem 501 gtp 1 vlan 777 ipktrule 10
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-501-gponport/bridge
```

Packet rule types

Packet rules types on the MXK-194/198:

- *bridgeinsertpppoevendortag*

PPPoE with intermediate agent (bridgeinsertpppoevendortag) on page 245

- *bridgeinsertoption82*

  Insert DHCP option 82 parameter. See *Option 82 DHCP on bridge packet rule (bridgeinsertoption82)* on page 253

- *bridgedhcprelay*

  Enables DHCP relay. *DHCP on bridge packet rules (DHCP relay, and Forbid OUI)* on page 260

- *bridgeforbidoui*

  Forbid OUI. See *DHCP on bridge packet rules (DHCP relay, and Forbid OUI)* on page 260

- *destmacswapdynamic*

  destmacswapstatic

  See *Destination MAC swapping (dstmacswapdynamic and dstmacswapstatic)* on page 265.

Configure packet-rule-records

Creating a packet rule group index with packet rule records

1. Use the **rule add** command to add a rule type to a group and member index and the parameter(s) which define that rule type.

   This example creates a packet-rule-group index with two members.

   The **dstmacswappingstatic** rule shown requires a parameter which is a MAC address. Entering *ipktrule* will enter the rules 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)
```

   If needed, add another rule to the group index.

```
zSH> rule add bridgedhcprelay 3/1 20
Created packet-rule-record 3/1 (bridgedhcprelay)
```

   Display the **packet-rule-group** with members.

```
zSH> rule show
Group/Member                                  Type Value(s)
-----------------------------------------------------------------------
       2/1                        dstmacswapstatic 08:00:20:bc:8b:8c
       3/1                        bridgedhcprelay 20
2 record(s) found
```

2. Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

3 Create bridges and include the IP packet rule groups.

```
zSH> bridge add 1-1-1-4/gpononu gem 501 gtp 1 downlink vlan 777 ipktrule 3
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-501-gponport/bridge

zSH> bridge add 1-1-2-0/eth uplink vlan 888 epktrule 2
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record 1-1-2-0-eth-888/bridge
Bridge-path added successfully
```

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
      10/1                        bridgedhcprelay 20
      10/2                   bridgeinsertoption82 circuitIDExample
      10/3                   bridgeinsertoption82 circuitIDExample remoteIDExample
4 record(s) found
```

PPPoE with intermediate agent (bridgeinsertpppoevendortag)

PPP headend servers (also known as Broadband Remote Access Servers or BRAS) authenticate and manage PPP connections.

TR-101 defines information which is entered into the packets when creating Point-to-Point Protocol over Ethernet connection through an Intermediate Agent (PPPoE Intermediate Agent).

Figure 21: PPPoE with intermediate agent

The MXK-194/198 is capable of being an intermediate agent in a PPPoE (point-to-point protocol over Ethernet) scenario as shown in Figure 21.

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.

PADI

During the discovery process, the PPPoE client (host) broadcasts a request by transmitting PPPoE Active Discovery Initiation (PADI) packets. 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.

PADS

The MXK-194/198 automatically inserts slot, port, SLAN/VLAN information into PPPoE packets that transits a MXK-194/198 bridge interface. The MXK-194/198 can also be configured to insert a customized string into the vendor-specific portion of the PPPoE packet when receiving a PPPoE Active Discovery Initiation (PADI) packet or a PPPoE Active Discovery Request (PADR) packet.

The customized string is entered into the **packetRuleValue** field of the **rule add** command.

The MXK-194/198 supports two ways to configure the packetRuleValue for the for the *bridgeinsertpppoevendortag* rule type. The first is without macro defined strings, see *PPPoE with intermediate agent configuration without macro defined strings* on page 247. The second is with macro defined strings, see *PPPoE with intermediate agent configuration with macro defined strings* on page 249.

Without macro defined strings, PPPoE behavior prepends as much text of the custom string as will fit in the field (from 0 to 48 characters) and the output text is truncated if required to fit into the **packetRuleValue** field.

PPPoE with intermediate agent configuration without macro defined strings

The customized identification string is 0 to 48 characters. The inserted information is TR-101 compliant and formatted as:

`<`***`customstring`***`> eth slot/port [[:stagID]:vlan-tag]`*slot/port SLAN and VLAN is default information automatically inserted into the packet*

The structure of the rule is that if a custom string is entered, that string, and only that string, will be entered in the packet. If a custom string is not entered, the eth slot/port [[:stadID] :vlan-tag] is entered.

The *slot/port* identifies the ingress *slot/port* on the MXK-194/198 where the packet was received. If the bridge is configured with a VLAN or SLAN tag, the VLAN/SLAN tag is also added to the packet.

When the **packetRuleValue** field is blank or contains a text string without a dollar sign, the **packetRuleValue** field is processed as shown in *Creating a packet rule for bridgeinsertpppoevendortag for default information* on page 247.

Creating a packet rule for bridgeinsertpppoevendortag for default information

Create a **packet-rule-record** with for default information.

1. Create the *bridgeinsertpppoevendortag* filter for default information.

```
zSH> rule add bridgeinsertpppoevendortag 1/1 ""
Created packet-rule-record 1/1 (bridgeinsertpppoevendortag)
```

2. Verify the **packet-rule-record**.

```
zSH> rule show
Group/Member                                    Type Value(s)
-----------------------------------------------------------------------
       1/1             bridgeinsertpppoevendortag
1 record(s) found
```

3. Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

4 Add the packet rule to a downlink bridge.

```
zSH> bridge add 1-1-1-4/gpononu gem 701 gtp 1 downlink vlan 101 ipktrule 1
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge
```

Applying the filter to this bridge causes the *eth slot/port* [[:*stagID*]:*vlan-tag*]to be inserted into the packets for PPPoE session establishment.

Note: For configurations with bridge intralinks or subtended SLMS devices, ensure that the PPPoE intermediate agent feature is enabled on only the subtended devices, or the downlink, or the TLS bridges.

Creating a packet rule for bridgeinsertpppoevendortag rule with custom string

1 Enter the **rule add** command with group/member index and custom string.

```
zSH> rule add bridgeinsertpppoevendortag 2/1 test_260
Created packet-rule-record 2/1 (bridgeinsertpppoevendortag)
```

2 Verify the rule.

```
zSH> rule show
Group/Member                                    Type Value(s)
-----------------------------------------------------------------------
       1/1             bridgeinsertpppoevendortag
       2/1             bridgeinsertpppoevendortag test_260
2 record(s) found
```

3 Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

4 Apply the packet rule to a downlink bridge.

```
zSH> bridge add 1-1-1-4/gpononu gem 701 gtp 1 downlink vlan 101 ipktrule 2
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge
```

Applying the filter to this bridge causes the custom string *test_mxk* to be inserted into the packets for PPPoE session establishment.

Deleting a packet-rule-record

Use the **rule delete** command to delete a packet rule.

```
zSH> rule delete 1/1
packet-rule-record 1/1 Delete complete
```

PPPoE with intermediate agent configuration with macro defined strings

The MXK-194/198 can be configured to insert a customized string into the vendor-specific portion of the PPPoE packet when receiving a PPPoE Active Discovery Initiation (PADI) packet or a PPPoE Active Discovery Request (PADR) packet.

If the **packetRuleValue** field contains one or more dollar sign ($) characters, the vendor tag text that would have been supplied is replaced by the contents of the field as follows:

This section discusses how to insert customized strings with the use of supported macro formats as shown in Table 6.

- When a dollar sign character is encountered, the text following the dollar sign is compared to Table 6.
- If no match is found the dollar sign character is replaced with the text "Unknown".
- If a match is found the dollar sign character and the associated text is replaced by the text indicated.
- The macro name and abbreviations are both case sensitive.
- The $macro strings may be imbedded in literal text. This text is copied to the output without change.

- The supported macro formats may be entered in the text as either $macroname or $abbreviation. Thus $SystemName and $NM give the same result, which is to substitute the system name from the **system 0** profile.

Some of the macros vary in effect depending on the value they are intended to display.

- $Gem and $Onu IDs are displayed or not depending on whether or not they have a non-zero value.
- $Vlan displays -SLAN-VLAN if the SLAN is non-zero, -VLAN if the -SLAN is zero but the VLAN is non-zero, or nothing if they are both zero.
- $VC displays -vpi-vci if either value is non-zero and nothing if they are both zero.

Note: Macro names are case sensitive.

Table 6: Supported macro formats for macro defined strings

Macro name	Abbreviation	Varies	Result
$SystemName	NM	NM	**sysname** from the **system 0** profile.
$SystemIP	IP	No	**ipaddress** address from the **system 0** profile.
$IfName	IF	IF	**ifName** from the bridge **IfTranslat**e profile.
$Address	AD	No	*shelf-slot-port-subport-type* of the underlying physical interface. Where the interface is a GPON OLT interface the type is given as *gponport* and the subport is the GEM port.
$Shelf	SH	No	Shelf (currently always 1).
$Slot	SL	No	*slot* from the **IfTranslate** profile of the underlying physical interface.
$Port	PT	No	*port* (see $Slot).
$SubPort	SP	No	*subport* (see $Slot.) For GPON this is the GEM port
$Gem	GM	Yes	-GEMPort (or nothing)
$Onu	ON	Yes	-ONUnumber (or nothing)
$Type	TY	No	Type (for GPON this is *gponport*).
$Vlan	VN	Yes	-SLAN-VLAN (or -VLAN or nothing).
$Svlan	SV	No	SLAN
$Cvlan	CV	No	VLAN
$Vc	VC	Yes	-VPI-VCI (or nothing)

Table 6: Supported macro formats for macro defined strings (Continued)

Macro name	Abbreviation	Varies	Result
$Vpi	VP	No	-VPI
$Vci	VI	No	-VCI
$Null	NL	No	Nothing (used to change PPPoE handling of constant strings).

Creating a packet rule for bridgeinsertpppoevendortag using macro names

Create a **packet-rule-record** using macro names to create a user-defined string. Strings created with macros, including the information pulled in by the macro, are limited to 48 characters.

1. To create a string with macro names that includes *shelf/slot/port/subport*, VLAN ID, and SLAN ID enter:

```
zSH> rule add bridgeinsertpppoevendortag 3/1
$SystemName$Shelf$Slot$Port$Subport$Vlan$Svlan
Created packet-rule-record 3/1 (bridgeinsertpppoevendortag)
```

2. Verify the **packet-rule-record**.

```
zSH> rule show
Group/Member                                Type Value(s)
-----------------------------------------------------------------------
       3/1            bridgeinsertpppoevendortag
$SystemName$Shelf$Slot$Port$Subport$Vlan$Svlan
1 record(s) found
```

3. Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

4. Apply the *bridgeinsertpppoevendortag* rule to the downlink bridge.

```
zSH> bridge add 1-1-1-4/gpononu gem 701 gtp 1 downlink vlan 101 ipktrule 3
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge
```

The ifName (typically *shelf/slot/port/subport*, and the VLAN ID and SLAN ID configured on the bridge will be inserted into the packets for PPPoE session establishment.

Deleting a packet-rule-record

Use the **rule delete** command to delete a packet rule.

```
zSH> rule delete 3/1
packet-rule-record 3/1 Delete complete
```

Creating a packet rule for bridgeinsertpppoevendortag rule using macro names and text

You can create a *bridgeinsertpppoevendortag* filter that combines macro names and text.

1. To create a string with macro names and text, in this case *oakland* and *system name*, enter

```
zSH> rule add bridgeinsertpppoevendortag 4/1 oakland$IfName$Vlan$Svlan
Created packet-rule-record 4/1 (bridgeinsertpppoevendortag)
```

2. Verify the **packet-rule-record**.

```
zSH> rule show
Group/Member                                    Type Value(s)
-----------------------------------------------------------------------
       4/1             bridgeinsertpppoevendortag oakland$IfName$Vlan$Svlan
1 record(s) found
```

3. Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

4 Apply the packet rule to the downlink bridge.

```
zSH> bridge add 1-1-1-4/gpononu gem 701 gtp 1 downlink vlan 101 ipktrule 4
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge
```

Applying the filter to this bridge causes the custom string to be inserted into the packets for PPPoE session establishment.

Deleting a packet-rule-record

Use the **rule delete** command to delete a packet rule.

```
zSH> rule delete 4/1
packet-rule-record 4/1 Delete complete
```

Option 82 DHCP on bridge packet rule (bridgeinsertoption82)

This section covers the two methods used to configure the *bridgeinsertoption82* rule type and includes:

- *Option 82 for DHCP relay overview* on page 253
- *Option 82 DHCP on bridge packet rule (bridgeinsertoption82) configuration without macros* on page 254
- *Option 82 DHCP on bridge packet rule (bridgeinsertoption82)configuration with macros* on page 255

Option 82 for DHCP relay overview

When acting as a DHCP relay agent, the MXK-194/198 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.

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 is it 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.

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

Option 82 DHCP on bridge packet rule (bridgeinsertoption82) configuration without macros

The default information inserted into the packet during the DHCP discovery process is formatted as:

```
System 0_ip:IfName
```

The systemIP address is taken from the IP address configured in the **system 0** profile. If the IP address is not defined in the system 0 profile, *0.0.0.0* is inserted.

Creating a packet rule for bridgeinsertoption82 without macro defined strings

1. Create the *bridgeinsertoption82* filter for default information.

```
zSH> rule add bridgeinsertoption82 1/1
Created packet-rule-record 1/1 (bridgeinsertoption82)
```

2. Verify the rule.

```
zSH> rule show
Group/Member                                    Type Value(s)
-----------------------------------------------------------------------
       1/1                    bridgeinsertoption82
1 record(s) found
```

3. To specify the first **packetRuleType**:

```
zSH> rule add bridgeinsertoption82 2/1 oakland
Created packet-rule-record 2/1 (bridgeinsertoption82)
```

4 Verify the rule.

```
zSH> rule show
Group/Member                                  Type Value(s)
----------------------------------------------------------------------
       1/1                   bridgeinsertoption82
       2/1                   bridgeinsertoption82 oakland
2 record(s) found
```

5 To specify only the second **packetRuleType**:

```
zSH> rule add bridgeinsertoption82 3/1 "" 510-555-1111
Created packet-rule-record 3/1 (bridgeinsertoption82)
```

6 Verify the rule.

```
zSH> rule show
Group/Member                                  Type Value(s)
----------------------------------------------------------------------
       1/1                   bridgeinsertoption82
       2/1                   bridgeinsertoption82 oakland
       3/1                   bridgeinsertoption82  510-555-1111
3 record(s) found
```

7 To specify both values:

```
zSH> rule add bridgeinsertoption82 4/1 oakland 510-555-1111
Created packet-rule-record 4/1 (bridgeinsertoption82)
```

8 Verify the rule.

```
zSH> rule show
Group/Member                                  Type Value(s)
----------------------------------------------------------------------
       1/1                   bridgeinsertoption82
       2/1                   bridgeinsertoption82 oakland
       3/1                   bridgeinsertoption82  510-555-1111
       4/1                   bridgeinsertoption82 oakland 510-555-1111
4 record(s) found
```

Option 82 DHCP on bridge packet rule (bridgeinsertoption82)configuration with macros

This section discusses how to insert customized strings with the use of supported macro formats as shown in Table 6.

If the **packetRuleValue** field contains one or more dollar sign ($) characters, the vendor tag text that would have been supplied is replaced by the contents of the field as follows:

- When a dollar sign character is encountered, the text following the dollar sign is compared to Table 6.

- If no match is found the dollar sign character is replaced with the text "Unknown".
- If a match is found the dollar sign character and the associated text is replaced by the text indicated.
- The macro name and abbreviations are both case sensitive.
- The $macro strings may be imbedded in literal text. This text is copied to the output without change.
- The supported macro formats may be entered in the text as either $macroname or $abbreviation. Thus $SystemName and $NM give the same result, which is to substitute the system name from the **system 0** profile.

Some of the macros vary in effect depending on the value they are intended to display.

- $Gem and $Onu IDs are displayed or not depending on whether or not they have a non-zero value.
- $Vlan displays -SLAN-VLAN if the SLAN is non-zero, -VLAN if the -SLAN is zero but the VLAN is non-zero, or nothing if they are both zero.
- $VC displays -vpi-vci if either value is non-zero and nothing if they are both zero.

Note: Macro names are case sensitive.

Table 7: Supported macro formats for macro defined strings

Macro name	Abbreviation	Varies	Result
$SystemName	NM	NM	**sysname** from the **system 0** profile.
$SystemIP	IP	No	**ipaddress** address from the **system 0** profile.
$IfName	IF	IF	**ifName** from the bridge **IfTranslat**e profile.
$Address	AD	No	*shelf-slot-port-subport-type* of the underlying physical interface. Where the interface is a GPON OLT interface the type is given as *gponport* and the subport is the GEM port.
$Shelf	SH	No	Shelf (currently always 1).
$Slot	SL	No	*slot* from the **IfTranslate** profile of the underlying physical interface.
$Port	PT	No	*port* (see $Slot).
$SubPort	SP	No	*subport* (see $Slot.) For GPON this is the GEM port
$Gem	GM	Yes	-GEMPort (or nothing)

Table 7: Supported macro formats for macro defined strings

Macro name	Abbreviation	Varies	Result
$Onu	ON	Yes	-ONUnumber (or nothing)
$Type	TY	No	Type (for GPON this is *gponport*).
$Vlan	VN	Yes	-SLAN-VLAN (or -VLAN or nothing).
$Svlan	SV	No	SLAN
$Cvlan	CV	No	VLAN
$Vc	VC	Yes	-VPI-VCI (or nothing)
$Vpi	VP	No	-VPI
$Vci	VI	No	-VCI
$Null	NL	No	Nothing (used to change PPPoE handling of constant strings).

Creating a packet rule for bridgeinsertoption82 with macro defined strings

Create a **packet-rule-record** using macro names to create a user-defined string. Strings created with macros, including the information pulled in by the macro, are limited to 48 characters.

1 To create a string for the first **packetRuleType** field:

a To create a string that includes system IP address, IfName (typically *shelf/slot/port/subport*), and VLAN ID for the first **packetRuleType** field, enter:

```
zSH> rule add bridgeinsertoption82 1/1 $SystemIP$IfName$Vlan
Created packet-rule-record 1/1 (bridgeinsertoption82)
```

The $SystemIP macro looks in the **system 0** profile for the IP address and to the bridge configuration for the rest of the information.

View the **system 0** profile.

```
zSH> get system 0
system  0
syscontact: ----------->  {Zhone Global Services and Support 7001
Oakport Street Oakland Ca. (877) Zhone20 (946-6320) Fax (510)777-7113
support@zhone.com}
sysname: -------------->  {mx198-10GE-TOP}
syslocation: ---------->  {Oakland}
enableauthtraps: ------>  {disabled}
setserialno: ---------->  {0}
zmsexists: ------------>  {true}
zmsconnectionstatus: -->  {inactive}
zmsipaddress: --------->  {172.16.89.220}
configsyncexists: ----->  {false}
configsyncoverflow: --->  {false}
```

```
configsyncpriority: --->  {high}
configsyncaction: ----->  {noaction}
configsyncfilename: --->  {192.168.254.205_4_1307641204880}
configsyncstatus: ----->  {synccomplete}
configsyncuser: ------->  {zmsftp}
configsyncpasswd: ----->  ** private **
numshelves: ----------->  {1}
shelvesarray: --------->  {}
numcards: ------------->  {1}
ipaddress: ------------>  {192.168.254.205}
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)}
```

b Verify the **packet-rule-record**.

```
zSH> rule show
Group/Member                                    Type Value(s)
-----------------------------------------------------------------------
       1/1                   bridgeinsertoption82 $SystemIP$IfName$Vlan
1 record(s) found
```

c Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

d Add the *bridgeinsertoption82* rule to the downlink bridge.

```
zSH> bridge add 1-1-1-4/gpononu gem 701 gtp 1 downlink vlan 101 ipktrule 1
```

```
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge
```

Applying the filter to this bridge causes the custom strings to be inserted into the packets during the DHCP discovery process.

2 To create a string for the second **packetRuleType** field:

a To create a string for only the second **packetRuleType** field of the *bridgeinsertoption82* rule:

```
zSH> rule add bridgeinsertoption82 2/1 "" $SystemName
Created packet-rule-record 2/1 (bridgeinsertoption82)
```

b Verify the **packet-rule-record**.

```
zSH> rule show
Group/Member                                    Type Value(s)
-----------------------------------------------------------------------
       1/1                   bridgeinsertoption82 $SystemIP$IfName$Vlan
       2/1                   bridgeinsertoption82  $SystemName
2 record(s) found
```

3 To create a rule for the first and the second **packetRuleType** fields:

a To create a string for both the first and the second **packetRuleType** fields of the *bridgeinsertoption82* rule:

```
zSH> rule add bridgeinsertoption82 3/1 $SystemName $SystemIP$IfName$Vlan
Created packet-rule-record 3/1 (bridgeinsertoption82)
```

b Verify the **packet-rule-record**.

```
zSH> rule show
Group/Member                                    Type Value(s)
-----------------------------------------------------------------------
       1/1                   bridgeinsertoption82 $SystemIP$IfName$Vlan
       2/1                   bridgeinsertoption82  $SystemName
       3/1                   bridgeinsertoption82 $SystemName
$SystemIP$IfName$Vlan
3 record(s) found
```

4 Add a packet rule for *bridgeinsertoption82* to a downlink bridge.

```
zSH> bridge add 1-1-1-5/gpononu gem 901 gtp 1 downlink vlan 1001 ipktrule 3
Adding bridge on 1-1-1-5/gpononu
Created bridge-interface-record 1-1-5-901-gponport/bridge
```

Applying the filter to this bridge causes the custom strings to be inserted into the packets during the DHCP discovery process.

Deleting a packet-rule-record

Use the **rule delete** command to delete a packet rule.

```
zSH> rule delete 1/1
```

```
packet-rule-record 1/1 Delete complete
```

DHCP on bridge packet rules (DHCP relay, and Forbid OUI)

This section describes:

- DHCP relay, page 260
- Forbid OUI, page 264

DHCP relay

Add the DHCP packet rule options using the **rule add** command to specify the packet rule option and which **packet-rule-record** group.

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, page 261.

```
zSH> dhcp-relay add 20 11.1.1 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: group: 20, index: 1

zSH> rule add bridgedhcprelay 10/1 20
Created packet-rule-record 10/1 (bridgedhcprelay)
```

Verify the rule.

```
zSH> rule show
Group/Member                                Type Value(s)
-----------------------------------------------------------------------
      10/1                        bridgedhcprelay 20
1 record(s) found
```

Bridge configuration with DHCP relay

The MXK-194/198 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. See *DHCP on bridge packet rules (DHCP relay, and Forbid OUI)* on page 260.

Figure 22 illustrates the traffic flow when the MXK-194/198 is configured with a bridge to support DHCP relay.

Figure 22: Bridge supported DHCP relay

Configuring bridges to support DHCP relay

This procedure describes how to configure bridges on the MXK-194/198 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-194/198 with a route available to the DHCP server.

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. Add an IP interface with a route to the DHCP server.

```
zSH> interface add 1-1-2-0/eth 11.1.1.198/24
Created ip-interface-record 1-1-2-0-eth/ip
```

2. 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: group: 20, index: 2
```

3 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)
```

4 Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

5 Create bridge and include IP packet rule group.

```
zSH> bridge add 1-1-1-4/gpononu gem 501 gtp 1 downlink vlan 700 ipktrule 10
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-501-gpononu-700/bridge
```

6 You can verify the information in the profiles:

a Verify the **dhcp-server-subnet** subnet group.

```
   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}
```

b Verify the **bridge-interface-record** contains the packet rule group:

```
   zSH> get bridge-interface-record 1-1-4-501-gpononu-700/bridge
bridge-interface-record 1-1-4-501-gpononu-700/bridge
vpi: --------------------------------->  {0}
vci: --------------------------------->  {0}
vlanId: ------------------------------->  {840}
stripAndInsert: ---------------------->  {false}
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}
vlan-xlate-from: ---------------------->  {0}
slan-xlate-from: ---------------------->  {0}
```

c 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
```

d 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: ->  {}
```

Forbid OUI

The *bridgeforbidoui* rule is filtering based on Organizational Unique Indentifer (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).

Configuring bridges to support forbid OUI

7 Add the bridgeforbidoui rule to the IP packet rule group.

```
zSH> rule add bridgeforbidoui 10/1 00:aa:bb
Created packet-rule-record 10/1 (bridgeforbidoui)
```

Packets from a device with a MAC address which begins with “00:aa:bb” the hexadecimal vendor code (OUI — Organizational Unique Identifier) will be blocked.

8 Verify the rule.

```
zSH> rule show
Group/Member                                    Type Value(s)
-----------------------------------------------------------------------
      10/1                          bridgeforbidoui 00:aa:bb
1 record(s) found
```

9 Create a bridge on upstream ethernet port.

```
zSH> bridge add 1-1-2-0/eth tls vlan 50 tagged
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record 1-1-2-0-eth-50/bridge
Bridge-path added successfully
```

10 Create a GPON traffic profile.

```
zSH> new gpon-traffic-profile 1

gpon-traffic-profile 1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512000
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

11 Create bridge and include IP packet rule group.

```
zSH> bridge add 1-1-1-4/gpononu gem 501 gtp 1 downlink vlan 700 ipktrule 10
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-501-gpononu-700/bridge
```

Destination MAC swapping (dstmacswapdynamic and dstmacswapstatic)

The destination MAC swapping feature provides a security enhancement which prevents port-to-port communications between users sharing a VLAN for Internet access when the user-to-user traffic spans multiple MXK-194/198 shelves.

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-194/198 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-194/198 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-194/198 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 when 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-194/198 and the DHCP server.

- Static user-specified entry

  The MXK-194/198 inserts the user-specified valid 6-byte hexadecimal destination MAC address into unicast frames not matching the static entry.

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 destination 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 next 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)

zSH> bridge add 1-1-1-4/gpononu gem 701 gtp 1 downlink vlan 101 ipktrule 1
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge
```

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)

zSH> bridge add 1-1-1-4/gpononu gem 701 gtp 1 downlink vlan 101 ipktrule 2
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge
```

Advanced bridging topics

This section describes:

- Bridges with IGMP, page 267
- “Denial of Service” prevention, page 272
- Rapid Spanning Tree Protocol (RSTP), page 272

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-194/198 supports IGMPv1, v2, IGMP snooping and IGMP proxy reporting.

Figure 23: IGMP video

The following video bridge example describes a video bridge on an uplink GigE interface and the bridge path on that interface. The GPON downlink bridge places the number of video streams and the multicast control list.

For the uplink bridge enter:

```
zSH> bridge add 1-1-5-0/eth uplink vlan 777
Adding bridge on 1-1-5-0/eth
Created bridge-interface-record 1-1-5-0-eth-777/bridge
Bridge-path added successfully
```

For the uplink bridge path, add a bridge path and a multicast aging period and IGMP query interval.

```
zSH> bridge-path add 1-1-5-0-eth-777/bridge vlan 777 default mcast 90 igmptimer 30
Bridge-path added successfully
```

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 signals.

Specifying a multicast control list of *0* allows all IP multicasts.

```
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> bridge add 1-1-1-4/gpononu gem 701 gtp 1 downlink vlan 101 video 1/6
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-4-701-gponport/bridge
```

Delete the bridging configuration if necessary.

```
zSH> bridge delete 1-1-5-0-eth-777/bridge
Bridge-path deleted successfully
1-1-5-0-eth-777/bridge delete complete
```

```
zSH> bridge delete 1-1-4-701-gponport/bridge
1-1-4-701-gponport/bridge delete complete
```

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-194/198 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-194/198 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 an 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 1-1-4-701-gponport/bridge

bridge-interface-record  1-1-4-701-gponport/bridge
vpi: --------------------------------->  {0}
vci: --------------------------------->  {0}
vlanId: ------------------------------->  {101}
stripAndInsert: ---------------------->  {true}
customARP: --------------------------->  {false}
filterBroadcast: --------------------->  {false}
learnIp: ----------------------------->  {true}
learnUnicast: ------------------------->  {true}
maxUnicast: -------------------------->  {5}
learnMulticast: ---------------------->  {true}
forwardToUnicast: -------------------->  {false}
forwardToMulticast: ------------------->  {false}
forwardToDefault: -------------------->  {true}
bridgeIfCustomDHCP: ------------------->  {false}
bridgeIfIngressPacketRuleGroupIndex: ->  {2}
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}
vlan-xlate-from: --------------------->  {0}
slan-xlate-from: --------------------->  {0}
```

For the downlink bridge, the **forwardToMulticast** setting is *false* and the **learnMulticast** setting is *true*.

```
zSH> get bridge-interface-record 1-1-4-701-gponport/bridge

bridge-interface-record  1-1-4-701-gponport/bridge
vpi: --------------------------------->  {0}
vci: --------------------------------->  {0}
vlanId: ------------------------------->  {101}
stripAndInsert: ---------------------->  {true}
customARP: --------------------------->  {false}
filterBroadcast: --------------------->  {false}
learnIp: ----------------------------->  {true}
learnUnicast: ------------------------->  {true}
maxUnicast: -------------------------->  {5}
```

```
learnMulticast: ---------------------->  {true}
forwardToUnicast: -------------------->  {false}
forwardToMulticast: ------------------>  {false}
forwardToDefault: -------------------->  {true}
bridgeIfCustomDHCP: ------------------>  {false}
bridgeIfIngressPacketRuleGroupIndex: ->  {2}
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}
vlan-xlate-from: --------------------->  {0}
slan-xlate-from: --------------------->  {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), and is backward compatible with STP. 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 24: 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 known 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 bridges. The root port will receive the best BPDU from the root switch on a bridge.

In Figure 24, 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 Figure 24, 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 Figure 24, 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. To show RSTP information use the **stp-bridge show** command.

RSTP on uplinks

Rapid Spanning Tree Protocol (RSTP, IEEE 802.1W) is supported on upstream interfaces.

Note: Interface *1-1-1-0/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-194/198 upstream ports *1-1-4-0/eth* and *1-1-3-0/eth*.

```
zSH> stp-bridge add 1-1-4-0/eth uplink vlan 500
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-500/bridge
Bridge-path added successfully

zSH> stp-bridge add 1-1-3-0/eth uplink vlan 500
Adding bridge on 1-1-3-0/eth
Created bridge-interface-record 1-1-3-0-eth-500/bridge
Bridge-path added successfully
```

   Use **stp-bridge add** *interface/type* **uplink vlan** *x* <*tagged*> to add a VLAN interface to the upstream interface.

   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 entering **bridge show**.

2. Verify the bridges.

```
zSH> bridge show
     Orig
```

```
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------------------------
upl             Tagged  500    1/1/3/0/eth             1-1-3-0-eth-500/bridge        DIS  STP: ROOT
upl             Tagged  500    1/1/4/0/eth             1-1-4-0-eth-500/bridge        LRN  STP: DSNT
2 Bridge Interfaces displayed
```

Port *1-1-3-0* has been chosen as the root port, which is an active uplink port is receiving and forwarding packets. Port *1-1-4-0* is the alternate port, which is blocked and discarding packets.

The **stp-bridge show** command provides STP information on the *stp* bridges.

```
zSH> stp-bridge show
Bridge is running IEEE 802.1W RSTP
Bridge ID has priority 36000, address 00:01:47:27:14:86
Configured: hello=2, forward=15, max_age=20
Current root has priority 32768, address 00:04:96:19:22:f0
Cost of root path 20000
1 bridge(s) present first-> 1-1-4-0-eth-500:
       is a DESIGNATED PORT in FORWARDING state
Root bridge has priority 32768, address 00:04:96:19:22:f0
Designated bridge has priority 36000, address 00:01:47:27:14:86
Designated Port id is 128:128, root path cost is 20000
Timers: forward delay is 15, hello time is 2, message age is 1
sync: 0 synced: 0 reRoot: 0 rrWhile: 0 operEdge: 0 fdWhile: 0
learn: 1 forward: 1 agreed: 0 learning: 1 forwarding: 1 updtInfo: 0 selected: 1

1 bridge(s) present first-> 1-1-3-0-eth-500:
       is a ROOT PORT in FORWARDING state
Root bridge has priority 32768, address 00:04:96:19:22:f0
Designated bridge has priority 32768, address 00:04:96:19:22:f0
Designated Port id is 16385:144, root path cost is 0
Timers: forward delay is 15, hello time is 2, message age is 0
sync: 0 synced: 0 reRoot: 0 rrWhile: 15 operEdge: 0 fdWhile: 0
learn: 1 forward: 1 agreed: 0 learning: 1 forwarding: 1 updtInfo: 0 selected: 1
```

3 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 can be displayed with the **get stp-bind** <*profile-storage-key*> command, and can be changed using **update stp-bind** <*profile-storage-key*> command.

To view existing **stp-bind** profiles and verify the port priority, enter:

```
zSH> list stp-bind
stp-bind  1-1-4-0-eth/linegroup/0
stp-bind  1-1-3-0-eth/linegroup/0
2 entries found.

zSH> get stp-bind 1-1-4-0-eth/linegroup/0
stp-bind  1-1-4-0-eth/linegroup/0
portPriority: ->  {128}

zSH> get stp-bind 1-1-5-0-eth/linegroup/0
```

```
stp-bind  1-1-5-0-eth/linegroup/0
portPriority: ->  {144}
```

4 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}
```

5 Delete the **stp-bridge**(s) on the ports.

```
zSH> stp-bridge delete 1-1-3-0-eth-500/bridge
Bridge-path deleted successfully
1-1-3-0-eth-500/bridge delete complete
zSH>
zSH> stp-bridge delete 1-1-4-0-eth-500/bridge
Bridge-path deleted successfully
1-1-4-0-eth-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 Figure 25 for an RSTP rlink ring topology. In this example, there is the mixed use of MXK-194/198 and MXK in a network. Each MXK-194/198 and MXK has a bridge interface with the characteristics of an uplink bridge enabled on one 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 MXK-194/198 would be sent upstream towards the root switch out the interface A1.

Figure 25: RSTP rlink ring topology

Figure 25 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 MXK-194/198's root port interface A1.

Figure 26: RSTP rlink with a different downed link

Configuring RSTP rlinks

The configuration procedures for the RSTP rlink topologies are listed below.

Note: This example show RSTP rlinks configured on both uplink and intralink ports on the MXK-194/198 and MXK. You can also configure pure RSTP on the uplink port, and configure RSTP rlink on the intralink port.

1 As shown in Figure 25, on the MXK-194/198, to configure RSTP rlinks on uplink and intralink bridges, perform the following tasks:

a Create RSTP rlink on upstream port A1 (*1-1-4-0*) and intralink port A2 (*1-1-3-0*) with **stp-bridge add** *interface/type* **rlink vlan** *id* *<tagged>*.

```
zSH> stp-bridge add 1-1-4-0/eth rlink vlan 500
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-500/bridge
Bridge-path added successfully
Bridge-path added successfully

zSH> stp-bridge add 1-1-3-0/eth rlink vlan 500
Adding bridge on 1-1-3-0/eth
Created bridge-interface-record 1-1-3-0-eth-500/bridge
Bridge-path added successfully
Bridge-path added successfully
```

If the parameter *vlan id* is not specified, VLAN 0 is used. And if parameter *tagged* is not specified, the rlink bridge is considered a tagged bridge.

Verify the bridges.

```
zSH> bridge show vlan 500
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-----------------------------------------------------------------------------------------------------------
-
rlk             Tagged  500    1/1/3/0/eth              1-1-3-0-eth-500/bridge        FWD S VLAN 500 Default STP:
ROOT

                                                                                        S VLAN 500 Intralink
STP: ROOT
rlk             Tagged  500    1/1/4/0/eth               1-1-4-0-eth-500/bridge         FWD S VLAN 500 Intralink
STP: DSNT
2 Bridge Interfaces displayed
```

b To create the bridge paths for the rlink bridges, two **bridge-path add** *interface/type* **vlan** *vlanID* commands (**intralink** and **default**) may be manually provisioned for each rlink interface

Although the bridge-path is automatically created, you may manually provision the bridge-path with additional variables using the following **bridge-path add** command.

```
zSH> bridge-path add 1-1-3-0-eth-500/bridge vlan 500 default
Bridge-path added successfully
```

```
zSH> bridge-path add 1-1-3-0-eth-500/bridge vlan 500 intralink
Bridge-path added successfully

zSH> bridge-path add 1-1-4-0-eth-500/bridge vlan 500 default
Bridge-path added successfully

zSH> bridge-path add 1-1-4-0-eth-500/bridge vlan 500 intralink
Bridge-path added successfully
```

c Verify the bridge-paths.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
       500 1-1-4-0-eth-500/bridge              Intralink
       500 1-1-4-0-eth-500/bridge              Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
       500 1-1-3-0-eth-500/bridge              Intralink
       500 1-1-3-0-eth-500/bridge              Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
```

d Show the baseline of the system, enter:

```
zSH> bridge show vlan 500
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------------------------
-
rlk            Tagged  500    1/1/3/0/eth             1-1-3-0-eth-500/bridge       FWD S VLAN 500 Default STP:
ROOT

                                                                                      S VLAN 500 Intralink
STP: ROOT
rlk            Tagged  500    1/1/4/0/eth             1-1-4-0-eth-500/bridge        FWD S VLAN 500 Intralink
STP: DSNT
2 Bridge Interfaces displayed
```

Port A1 (*1-1-3-0*) has been chosen as the root port, which is an active uplink port in the forwarding state. Port A2 (*1-1-4-0*) 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
Bridge-path added successfully
Bridge-path added successfully

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
Bridge-path added successfully
Bridge-path added successfully
```

b Verify the bridge paths:

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
--------------------------------------------------------------------------------
       500 ethernet4-500/bridge                Intralink
       500 ethernet4-500/bridge                Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
       500 ethernet5-500/bridge                Intralink
       500 ethernet5-500/bridge                Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
```

c Show the baseline of the system, enter.

```
zSH> bridge show vlan 500
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                        St  Table Data
-----------------------------------------------------------------------------------------------------------
-
rlk            Tagged  500    1/1/3/0/eth              1-1-3-0-eth-500/bridge       FWD S VLAN 500 Default STP:
ROOT

                                                                                      S VLAN 500 Intralink
STP: ROOT
rlk            Tagged  500    1/1/4/0/eth              1-1-4-0-eth-500/bridge       FWD S VLAN 500 Intralink
STP: DSNT
2 Bridge Interfaces displayed
```

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 Figure 26, if the connection between the MXK-194/198 uplink port A1 to the root switch is broken, the intralink port A2 on the MXK-194/198 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 MXK-194/198, verify uplink port A1(*1-1-4-0*) is down, intralink port A2 (*1-1-3-0)* is in the forwarding state and takes over the role of root port, enter.

```
zSH> bridge show vlan 500
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                        St  Table Data
-----------------------------------------------------------------------------------------------------------
-
rlk            Tagged  500    1/1/3/0/eth              1-1-3-0-eth-500/bridge       FWD S VLAN 500 Default STP:
ROOT

                                                                                      S VLAN 500 Intralink
STP: ROOT
rlk            Tagged  500    1/1/4/0/eth              1-1-4-0-eth-500/bridge       FWD S VLAN 500 Intralink
STP: DSNT
2 Bridge Interfaces displayed
```

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 necessary, delete the STP bridge on the port.

a To delete the bridge on MXK-194/198, use **stp-bridge delete** *interface*/*type* command.

```
zSH> stp-bridge delete 1-1-4-0-eth-500/bridge
Bridge-path deleted successfully
1-1-4-0-eth-500/bridge delete complete
```

b To delete the bridge on MXK, use **stp-bridge delete** *interface*/*type* command.

```
zSH> stp-bridge delete ethernet4-500/bridge
Bridge-path deleted successfully
ethernet4-500/bridge delete complete
```

VLAN translation

For GPON line card, VLAN translation is performed on the ONU instead of on the MXK-194/198. VLAN translation on ONU is also referred to as CPE VLAN translation in this document.

Note: CPE VLAN translation is only supported in Dynamic OMCI.

Refer to *VLAN translation on ONU* on page 415 for detail.

MXK-194/198 bridging configurations

This section includes the following bridging topics:

- Configure a tagged uplink bridge with VLAN ID, page 282
- Configure tagged or untagged downlink bridges with GEM Port IDs, page 283
- Configure tagged downlink bridges on GPON, page 284
- Configure bridges using Q-in-Q (VLAN IDs and SLAN IDs), page 286
- Configure TLS bridges, page 293
- Configure TLS wire bridges, page 294
- TLS bridge parameters floodUnknown and floodMulticast, page 295
- Configure link aggregation bridges, page 298
- Configure bridge loop issue prevention, page 299

- Dynamic IP filtering on a bridge (Secure DHCP), page 300
- Broadcast suppression, page 302
- Upstream multicast video support, page 303
- Multicast VPN point-to-point service support on a wire bridge for GPON, page 304

Configure a tagged uplink bridge with VLAN ID

All uplink bridges on the MXK-194/198 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-194/198, 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 defaults* on page 222 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 bridging parameters.

Creating an uplink bridge

Create an uplink bridge on an uplink.

1 Create the uplink bridge, then verify the bridge.

```
zSH> bridge add 1-1-2-0/eth uplink vlan 555
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record 1-1-2-0-eth-555/bridge
Bridge-path added successfully

zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                        St  Table Data
-------------------------------------------------------------------------------------------
upl            Tagged  555    1/1/2/0/eth              1-1-2-0-eth-555/bridge        DWN S VLAN 555 default
1 Bridge Interfaces displayed
```

2 View the uplink bridge **bridge-interface-record** profile **stripAndInsert** parameter.

```
zSH> get bridge-interface-record 1-1-2-0-eth-555/bridge
bridge-interface-record  1-1-2-0-eth-555/bridge
vpi: ---------------------------------->  {0}
vci: ---------------------------------->  {0}
vlanId: ------------------------------->  {555}
stripAndInsert: ----------------------->  {false}
```

Creating this uplink bridge with the **bridge add** command inserts 555 into the **vlanId** parameter and sets the **stripAndInsert** parameter to *false* in the **bridge-interface-record** profile.

Configure tagged or untagged downlink bridges with GEM Port IDs

You can configure downlink bridges on the MXK-194/198 using the variables *tagged* or *untagged* depending on the downstream configuration and the downstream bridging behavior that you desire. See *Configuring a GPON untagged downlink VLAN bridge* on page 284 and *Configuring a GPON tagged downlink VLAN bridge* on page 285 for configuration procedures.

Untagged downlink bridges on GPON interfaces

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 towards 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 1-1-1-4/gpononu gem 510 gtp 1 downlink vlan 55
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-1-510-gponport/bridge
```

and

```
zSH> bridge add 1-1-1-4/gpononu gem 510 gtp 1 downlink vlan 55 untagged
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-1-510-gponport/bridge
```

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 a GPON untagged downlink VLAN bridge

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 1-1-1-4/gpononu gem 510 gtp 1 downlink vlan 100
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-1-510-gponport/bridge
```

This example creates an untagged downlink interface on the GPON port 1/ONU port 4/ GEM port 510 with a VLAN ID of *100*.

2. To verify the downlink bridge, enter **bridge show**.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------------------
dwn                100       1/1/1/4/gpononu1-1-1-510-gponport/bridge           DWN
1 Bridge Interfaces displayed
```

3. To view the **stripAndInsert** setting for the downlink bridge, enter.

```
zSH> get bridge-interface-record 1-1-1-510-gponport/bridge
bridge-interface-record  1-1-1-510-gponport/bridge
vpi: ---------------------------------->  {0}
vci: ---------------------------------->  {0}
vlanId: ------------------------------->  {100}
stripAndInsert: ----------------------->  {true}

...
```

The **vlanId** parameter is set to *100*, and the **stripAndInsert** parameter is set to *true*, meaning the VLAN ID *100* 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 GPON

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 a GPON tagged downlink VLAN bridge

1 Create a *tagged* downlink bridge with a VLAN ID.

```
zSH> bridge add 1-1-1-4/gpononu gem 510 gtp 1 downlink vlan 555 tagged
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-1-510-gponport-555/bridge
```

2 To display the tagged downlink bridge, enter **bridge show**.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------------
dwn            Tagged  555    1/1/1/4/gpononu 1-1-1-510-gponport-555/bridge        DWN
1 Bridge Interfaces displayed
```

3 View the **stripAndInsert** parameter of the **bridge-interface-record**.

```
zSH> get bridge-interface-record 1-1-1-510-gponport-555/bridge
bridge-interface-record  1-1-1-510-gponport-555/bridge
vpi: ---------------------------------->  {0}
vci: ---------------------------------->  {0}
vlanId: ------------------------------->  {555}
stripAndInsert: ----------------------->  {false}

...
```

The VLAN ID parameter is set to *555*. 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 downlink bridges

Deleting the bridge automatically deletes the static bridge path.

Deleting an uplink bridge

4 Delete the uplink bridge.

```
zSH> bridge delete 1-1-4-0-eth-800/bridge vlan 800
Bridge-path deleted successfully
1-1-4-0-eth-800/bridge delete complete
```

Deleting a downlink bridge

Delete the downlink bridge.

Delete the downlink bridge.

```
zSH> bridge delete 1-1-1-510-gponport-555/bridge
1-1-1-510-gponport-555/bridge delete complete
```

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 *stagged* 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 *stagged*, 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-194/198 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-194/198 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 Queuing* on page 239.

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: ----------------------------> {501} SLAN ID
s-tagStripAndInsert: -----------------> {true} Specifies whether or not to strip and insert
s-tagOutgoingCOSOption: --------------> {s-tagdisable} 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
```

Q-in-Q bridging configurations

The MXK-194/198 supports two ways of configuring Q-in-Q in bridging. The first way uses the *tagged* variable and the second way uses the *stagged* variable. Some MXK-194/198 bridging configurations are from an *stagged* bridge to a *tagged* bridge (see *Tagged downlink to stagged uplink bridge*

configuration (tagged to stagged) on page 287), or from a *stagged* bridge to a *stagged* bridge (see *stagged intralink bridge and stagged uplink bridge (stag to stag)* on page 290).

These bridge types can go from uplink to downlink or from downlink to uplink depending on your configuration requirements.

Note: The initial software implementation of the MXK-194/198 does not support double-tagging an untagged Ethernet frame. This function will be supported in future releases.

Tagged downlink to stagged uplink bridge configuration (tagged to stagged)

Figure 27 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 *stagged* 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 GPON ONT to receive the VLAN ID and not the SLAN ID. Figure 27 shows a tagged downlink and stagged uplink bridging configuration.

Figure 27: Tagged downlink and stagged uplink example

Configuring a tagged downlink and stagged 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 VLAN ID 101 on the GPON GEM port:

```
zSH> bridge add 1-1-4-1/gpononu gem 501 gtp 1 downlink vlan 101 slan 501 tagged
Adding bridge on 1-1-4-1/gpononu
```

```
Created bridge-interface-record 1-1-4-501-gponport-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 **stripAndInser**t 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 1-1-4-501-gponport-101/bridge
bridge-interface-record  1-1-4-501-gponport-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: ------------------------------>  {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}
vlan-xlate-from: ---------------------->  {0}
slan-xlate-from: ---------------------->  {0}
```

2 To verify the bridge enter:

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-----------------------------------------------------------------------------------------------------
-----------
dwn             Tg  101/501    1/1/4/1/gpononu           1-1-4-501-gponport-101/bridge  DWN
1 Bridge Interfaces displayed
```

3 Create an *stagged* uplink bridge with a VLAN ID and a SLAN ID to match the downlink bridge:

```
zSH> bridge add 1-1-4-0/eth uplink vlan 101 slan 501 stagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-101-501/bridge
Bridge-path added successfully
```

Designating the uplink bridge as *stagged* does not strip or insert the either the VLAN ID or the SLAN ID. The **stripAndInser**t 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 1-1-4-0-eth-101-501/bridge
bridge-interface-record  1-1-4-0-eth-101-501/bridge
vpi: ---------------------------------->  {0}
vci: ---------------------------------->  {0}
vlanId: ------------------------------->  {101}
stripAndInsert: ----------------------->  {false}  no 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}  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}
vlan-xlate-from: ---------------------->  {0}
slan-xlate-from: ---------------------->  {0}
```

4 Verify the bridge:

```
zSH> bridge show
     Orig
```

```
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-----------------------------------------------------------------------------------------------------
-----------
dwn             Tg  101/501    1/1/4/1/gpononu           1-1-4-501-gponport-101/bridge  DWN
upl             ST  101/501    1/1/4/0/eth               1-1-4-0-eth-101-501/bridge     DWN S SLAN 501
VLAN 101 default
2 Bridge Interfaces displayed
```

5 Verify the bridge-path:

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
  101/501   1-1-4-0-eth-101-501/bridge          Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
```

stagged intralink bridge and stagged uplink bridge (stag to stag)

Figure 28 describes an stagged downlink to stagged uplink bridging configuration.

Figure 28: stagged intralink bridge and stagged uplink configuration

Configuring an stagged bridge on an MXK intralink and an stagged bridge on the MXK-194/198 uplink

1 Create an *stagged* intralink bridge with an SLAN ID and a VLAN ID from the MXK to the MXK-194/198.

```
zSH> bridge add 1-a-6-0/eth intralink vlan 101 slan 502 stagged
Adding bridge on 1-a-6-0/eth
Created bridge-interface-record ethernet6-101-502/bridge
Bridge-path added successfully
```

Designating the intralink bridge as *stagged* does not strip or insert the either the VLAN ID or the SLAN ID. The **stripAndInser**t 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 ethernet6-101-502/bridge
bridge-interface-record  ethernet6-101-502/bridge
```

```
vpi: ---------------------------------->  {0}
vci: ---------------------------------->  {0}
vlanId: ------------------------------->  {101}
stripAndInsert: ----------------------->  {false}  no strip and insert
customARP: ---------------------------->  {false}
filterBroadcast: ---------------------->  {false}
learnIp: ------------------------------>  {false}
learnUnicast: ------------------------->  {true}
maxUnicast: --------------------------->  {5}
learnMulticast: ----------------------->  {false}
forwardToUnicast: --------------------->  {true}
forwardToMulticast: ------------------->  {false}
forwardToDefault: --------------------->  {false}
bridgeIfCustomDHCP: ------------------->  {false}
bridgeIfIngressPacketRuleGroupIndex: ->  {0}
vlanIdCOS: ---------------------------->  {0}
outgoingCOSOption: -------------------->  {disable}
outgoingCOSValue: --------------------->  {0}
s-tagTPID: ---------------------------->  {0x8100}
s-tagId: ------------------------------>  {502}
s-tagStripAndInsert: ----------------->  {false}  no strip and insert
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}
vlan-xlate-from: ---------------------->  {0}
slan-xlate-from: ---------------------->  {0}
```

2 Verify the bridge:

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                      St  Table Data
-------------------------------------------------------------------------------------
int           ST  101/502    1/a/6/0/eth             ethernet6-101-502/bridge      DWN S SLAN 502 VLAN 101 Intralink
1 Bridge Interfaces displayed
```

3 Create an *stagged* uplink bridge with the same VLAN ID and SLAN ID as the intralink bridge from the MXK-194/198 to the MXK.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 101 slan 502 stagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-101-502/bridge
Bridge-path added successfully
```

Designating *stagged* sets the **stripAndInser**t 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 1-1-4-0-eth-101-502/bridge
bridge-interface-record  1-1-4-0-eth-101-502/bridge
vpi: ---------------------------------->  {0}
vci: ---------------------------------->  {0}
vlanId: ------------------------------->  {101}
stripAndInsert: ----------------------->  {false}  no strip and insert
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}
s-tagStripAndInsert: ----------------->  {false}  no strip and insert
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}
vlan-xlate-from: ---------------------->  {0}
slan-xlate-from: ---------------------->  {0}
```

4 Create an uplink bridge on the MXK to the IP network and designate *stagged* so that the VLAN ID and the SLAN ID are passed to the network.

```
zSH> bridge add 1-a-5-0/eth uplink vlan 101 slan 502 stagged
Adding bridge on 1-a-5-0/eth
Created bridge-interface-record ethernet5-101-502/bridge
Bridge-path added successfully
```

5 Verify the bridge paths:

```
zSH> bridge-path show
VLAN/SLAN   Bridge                                Address
```

```
-------------------------------------------------------------------------------
  101/502   ethernet6-101-502/bridge            Intralink
  101/502   ethernet5-101-502/bridge           Default, Age: 3600, MCAST Age: 150,
IGMP Query Interval: 70, Flap Mode: Default
```

Delete the uplink and intralink bridges

When necessary you can delete the uplink bridge on the MXK-194/198 and the intralink and uplink bridge on the MXK.

Deleting the uplink bridge

1 View the existing bridges on the MXK-194/198.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
upl           ST  101/502    1/1/4/0/eth             1-1-4-0-eth-101-502/bridge     UP  S SLAN 502 VLAN 101 default
1 Bridge Interfaces displayed
```

2 Delete the uplink bridge on the MXK-194/198.

```
zSH> bridge delete 1-1-4-0-eth-101-502/bridge vlan 101
Bridge-path deleted successfully
1-1-4-0-eth-101-502/bridge delete complete
```

Deleting the intralink bridge

1 View the existing bridge on the MXK.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
upl           ST  101/502    1/a/5/0/eth             ethernet5-101-502/bridge       DWN S SLAN 502 VLAN 101 default
int           ST  101/502    1/a/6/0/eth             ethernet6-101-502/bridge       DWN S SLAN 502 VLAN 101 Intralink
2 Bridge Interfaces displayed
```

2 Delete the intralink and uplink bridge on the MXK.

```
zSH> bridge delete ethernet5-101-502/bridge vlan 101
Bridge-path deleted successfully
ethernet5-101-502/bridge delete complete

zSH> bridge delete ethernet6-101-502/bridge vlan 101
Bridge-path deleted successfully
ethernet6-101-502/bridge delete complete
```

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-194/198 GPON interface.

```
zSH> bridge add 1-1-1-5/gpononu gem 510 gtp 2 tls vlan 900
Adding bridge on 1-1-1-5/gpononu
Created bridge-interface-record 1-1-1-510-gponport/bridge
Bridge-path added successfully
```

2 Create a TLS bridge on an Ethernet uplink interface.

```
zSH> bridge add 1-1-4-0/eth tls vlan 900
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth/bridge
```

3 Verify the bridges.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------------------
tls                900        1/1/4/0/eth              1-1-4-0-eth/bridge            DWN
tls                900        1/1/1/5/gpononu     1-1-1-510-gponport/bridge    DWN
2 Bridge Interfaces displayed
```

Deleting the TLS bridge configuration

1 Delete the bridge on the GPON interface.

```
zSH> bridge delete 1-1-1-510-gponport/bridge
1-1-1-510-gponport/bridge delete complete
```

2 Delete the bridge on the Ethernet uplink interface.

```
zSH> bridge delete 1-1-4-0-eth/bridge
Bridge-path deleted successfully
1-1-4-0-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 1-1-4-0/eth wire vlan 500
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth/bridge
```

2 Create the second wire bridge interface with the same VLAN ID.

```
zSH> bridge add 1-1-1-5/gpononu gem 650 gtp 2 wire vlan 500
Adding bridge on 1-1-1-5/gpononu
Created bridge-interface-record 1-1-1-650-gponport/bridge
```

3 View the wire bridges.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
wre                500        1/1/4/0/eth              1-1-4-0-eth/bridge            DWN
wre                500        1/1/1/5/gpononu     1-1-1-650-gponport/bridge   DWN
2 Bridge Interfaces displayed
```

If a VLAN ID is used for two wire bridges, the system prevents that VLAN ID from being used again.

```
zSH> bridge add 1-1-1-7/gpononu gem 543 gtp 2 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-194/198 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-194/198 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-194/198 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 1-1-1-4/gpononu gem 340 gtp 2 tls vlan 500
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-1-340-gponport/bridge
Bridge-path added successfully

zSH> get bridge-interface-record 1-1-1-340-gponport/bridge
bridge-interface-record  1-1-1-4-gponport/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}
vlan-xlate-from: ---------------------->  {0}
slan-xlate-from: ---------------------->  {0}
```

floodMulticast parameter

The **floodMulticast** parameter allows the MXK-194/198 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-194/198 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 1-1-1-4/gpononu gem 345 gtp 2 tls vlan 500
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-1-345-gponport/bridge
Bridge-path added successfully

zSH> get bridge-interface-record 1-1-1-345-gponport/bridge
bridge-interface-record  1-1-1-345-gponport/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}
vlan-xlate-from: ---------------------->  {0}
slan-xlate-from: ---------------------->  {0}
```

Configure link aggregation bridges

This section describes link aggregation bridging:

- Create a uplink bridge with link aggregation, page 298
- Create a TLS bridge with link aggregation, page 299

Ethernet ports can be bonded together into groups on the MXK-194/198. Link aggregation bridge type is supported on uplink bridges and TLS bridges.

See *Chapter 12, 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        partner: Sys            Pri     grp ID  admin   numLinks
-------------------------------------------------------------------------------
  1*  1   1-1-1-0       00:00:00:00:00:00       0x0     0x0      up      2
        links           slot    port    subport         admin
        ---------------------------------------------------------
         1-1-3-0         1       3       0               up
         1-1-4-0         1       4       0               up
```

2 Create an uplink bridge with link aggregation:

```
zSH> bridge add 1-1-1-0/linkagg uplink vlan 500 tagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record linkagg-1-1-500/bridge
Bridge-path added successfully
```

Since the Ethernet port designated for the uplink bridge is a member of a *linkagg* group, the bridge is created as a *linkagg* bridge type.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                      St  Table Data
-------------------------------------------------------------------------------------------
upl            Tagged  500    1/1/1/0/linkagg          linkagg-1-1-500/bridge        UP  S VLAN 500 default
1 Bridge Interfaces displayed
```

The bridge path is automatically created.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                                Address
-------------------------------------------------------------------------------
       500 linkagg-1-1-500/bridge             Default, Age: 3600, MCAST Age: 150,
IGMP Query Interval: 70, Flap Mode: Default
```

Deleting the link aggregation bridge

Delete a bridge with link aggregation enter:

```
zSH> bridge delete linkagg-1-1-500/bridge
Bridge-path deleted successfully
linkagg-1-1-500/bridge delete complete
```

Create a TLS bridge with link aggregation

Creating TLS bridges 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 Enter the bridge add command with *tls*.

```
zSH> bridge add 1-1-1-0/linkagg tls vlan 777
Adding bridge on 1-1-1-0/linkagg
Created bridge-interface-record linkagg-1-1/bridge
Bridge-path added successfully
```

2 Verify the bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                       St  Table Data
-------------------------------------------------------------------------------
tls               777       1/1/1/0/linkagg          linkagg-1-1/bridge           UP  D 00:04:96:19:22:f0
1 Bridge Interfaces displayed
```

Deleting the link aggregation TLS bridge

To delete the link aggregation bridge if necessary, enter:

```
zSH> bridge delete linkagg-1-1/bridge vlan 777
linkagg-1-1/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 1-1-5-0/eth uplink vlan 700
Adding bridge on 1-1-5-0/eth
Created bridge-interface-record 1-1-5-0-eth-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 1-1-5-0-eth-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 1-1-1-4/gpononu gem 345 gtp 1 downlink vlan 700
Adding bridge on 1-1-1-4/gpononu
Created bridge-interface-record 1-1-1-345-gponport/bridge

zSH> bridge add 1-1-1-3/gpononu gem 356 gtp 1 downlink vlan 700
Adding bridge on 1-1-1-3/gpononu
Created bridge-interface-record 1-1-1-356-gponport/bridge
```

4 To unblock a bridge that is blocked because of flap protection, use the **bridge unblock** *interface/type* command.

```
zSH> bridge unblock 1-1-1-345-gponport/bridge
```

Dynamic IP filtering on a bridge (Secure DHCP)

Note: MXK-194/198 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.

The MXK-194/198 enables secure DHCP settings on downlink bridges and *subscriber* facing TLS bridges 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 or TLS subscriber facing bridge interface and VLAN, the MXK-194/198 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 1-1-4-1/gpononu gem 546 gtp 1 downlink vlan 109 tagged secure
Adding bridge on 1-1-4-1/gpononu
Created bridge-interface-record 1-1-4-546-gponport-109/bridge

zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                        St
Table Data
-------------------------------------------------------------------------------------------
----------------------
dwn             Tagged  109    1/1/4/1/gpononu           1-1-4-546-gponport-109/bridge  UP
```

2. Display the **bridge-interface-record** for the configured downlink bridge to view the detailed bridge settings.

```
zSH> get bridge-interface-record 1-1-4-546-gponport-109/bridge
bridge-interface-record  1-1-4-546-gponport-109/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: ----------------------------->  {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: ------------>  {mac+ip}
bridgeIfDhcpLearn: ------------------->  {mac+ip}
mvrVlan: ----------------------------->  {0}
vlan-xlate-from: --------------------->  {0}
slan-xlate-from: --------------------->  {0}
```

Deleting the dynamic IP filter on a bridge

Delete the dynamic IP on a bridge filter if necessary.

```
zSH> bridge delete 1-1-4-546-gponport-109/bridge
1-1-4-546-gponport-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-record 1-1-4-546-gponport-109/bridge
bridge-interface-record  1-1-4-546-gponport-109/bridge
Please provide the following: [q]uit.
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}: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: ------------>  {mac+ip}:
bridgeIfDhcpLearn: ------------------->  {mac+ip}:
mvrVlan: ----------------------------->  {0}:
vlan-xlate-from: --------------------->  {0}:
slan-xlate-from: --------------------->  {0}:
...................
Save changes? [s]ave, [c]hange or [q]uit:s
```

Upstream multicast video support

The upstream multicast video feature are supported on some OMCI-based zNIDs (e.g. zNID-GPON-2520) for applications like video surveillance.

The upstream multicast video feature can be turned on on the OLT side by using the **update bridge-interface-record** command.

The following example turns on the upstream multicast video on a downlink bridge that was created on GEM port 901 for video services:

```
zSH> update bridge-interface-record
1-1-1-901-gponport-300/bridge
bridge-interface-record  1-1-1-901-gponport-300/bridge
Please provide the following: [q]uit.
vpi: ---------------------------------->  {0}:
vci: ---------------------------------->  {0}:
vlanId: ------------------------------->  {500}:
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}: true
bridgeIfEgressPacketRuleGroupIndex: -->  {0}:
bridgeIfTableBasedFilter: ------------>  {NONE(0)}:
bridgeIfDhcpLearn: -------------------->  {NONE(0)}:
mvrVlan: ------------------------------>  {0}:
vlan-xlate-from: ---------------------->  {0}:
slan-xlate-from: ---------------------->  {0}:
....................
Save changes? [s]ave, [c]hange or [q]uit:s
```

Multicast VPN point-to-point service support on a wire bridge for GPON

MXK-194/198 allows IP multicast through a wire bridge to support Multicast VPN (MVPN) point-to-point service for GPON. This allows multicast video conferencing equipment to communicate over a wire bridge. The MXK-194/198 does not apply any IGMP snooping or proxy functions on this type of service. Management of multicast streams is performed by the equipment within the customer's VPN.

MVPN point-to-point service is only supported on wire bridge for GPON. It is not supported for TLS or other bridge types for GPON line types. All other line types support MVPN point-to-point over wire bridges or TLS bridges.

MVPN point-to-point is automatically added to wire bridges:

```
zSH> bridge add 1-1-1-502/gponport gtp 1 wire vlan 101
tagged
Adding bridge on 1-1-1-502/gponport
Created bridge-interface-record 1-1-1-502-gponport-101/
bridge
```

Administrative commands

The MXK-194/198 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**

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
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-----------------------------------------------------------------------------------------------------------
-------
dwn             Tagged  998    1/1/1/1/gpononu           1-1-1-701-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/2/gpononu           1-1-1-702-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/3/gpononu           1-1-1-703-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/4/gpononu           1-1-1-704-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/5/gpononu           1-1-1-705-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/6/gpononu           1-1-1-706-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/7/gpononu           1-1-1-707-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/8/gpononu           1-1-1-708-gponport-998/bridge  UP
dwn             Tagged  109    1/1/4/1/gpononu           1-1-4-546-gponport-109/bridge  UP
upl            Tagged  998    1/1/8/0/eth              1-1-8-0-eth-998/bridge        UP  S VLAN 998 default
upl            Tagged  840    1/1/9/0/eth              1-1-9-0-eth-840/bridge        UP  S VLAN 840 default
11 Bridge Interfaces displayed
Aging counter:   7513
Renew failed:    0
Filter renewed:  0
Flap Suppresses: 0
```

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-----------------------------------------------------------------------------------------------------------
-------
dwn             Tagged  998    1/1/1/1/gpononu           1-1-1-701-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/2/gpononu           1-1-1-702-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/3/gpononu           1-1-1-703-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/4/gpononu           1-1-1-704-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/5/gpononu           1-1-1-705-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/6/gpononu           1-1-1-706-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/7/gpononu           1-1-1-707-gponport-998/bridge  UP
dwn             Tagged  998    1/1/1/8/gpononu           1-1-1-708-gponport-998/bridge  UP
dwn             Tagged  109    1/1/4/1/gpononu           1-1-4-546-gponport-109/bridge  UP
upl            Tagged  998    1/1/8/0/eth              1-1-8-0-eth-998/bridge        UP  S VLAN 998 default
```

```
upl            Tagged  840    1/1/9/0/eth              1-1-9-0-eth-840/bridge
UP  S VLAN 840 default
11 Bridge Interfaces displayed
```

9 VIDEO CONFIGURATION

This chapter explains how to configure the MXK-194/198 for video and includes the following sections:

- MXK-194/198 bridged video, page 307
- Configure bridged video on the MXK-194/198, page 308
- IGMP snooping with proxy reporting, page 313
- IGMP bridging statistics, page 322

MXK-194/198 bridged video

Video bridging enables video packets to be forwarded over bridges from a headend device down to a downstream device. In this case, the video travels from the source, or head-end device, using one video stream to passively traverse the MXK-194/198 backplane. This lowers the bandwidth requirements for video packets traversing the MXK-194/198.

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-194/198 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-194/198 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.

Table 9 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-194/198

This section describes how to configure the MXK-194/198 for video connections so that traffic passes between the MXK-194/198, the upstream video source, and the subscriber:

- Bridged video connection overview, page 308
- Configure a video connection on the MXK-194/198, page 309

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-194/198 for bridged video.

Configuring the MXK-194/198 for bridged video

1. Create an uplink bridge on a FE/GE uplink port.
   a. Create an uplink bridge with a VLAN ID.

      See *Creating an uplink bridge on an Ethernet uplink port for video* on page 309.

   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 an Ethernet uplink port for video* on page 309.
2. Create the multicast control lists.

   See *Creating multicast control lists* on page 310.

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 a GPON GEM port for video services* on page 311.

Configure a video connection on the MXK-194/198

You must create an uplink bridge on a FE/GE uplink and configure the bridge for video service and then create a downlink bridge to the subscriber.

Creating an uplink bridge on an Ethernet uplink port for video

You create a video bridge on the uplink by first creating an uplink bridge on an Ethernet 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 with a VLAN ID. Designating *tagged* will pass the VLAN ID up to the network.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 101 tagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-101/bridge
Bridge-path added successfully
```

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 1-1-4-0-eth-101/bridge
bridge-interface-record  1-1-4-0-eth-101/bridge
vpi: ---------------------------------->  {0}
vci: ---------------------------------->  {0}
vlanId: ------------------------------->  {101}
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}
vlan-xlate-from: ---------------------->  {0}
slan-xlate-from: ---------------------->  {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 add 1-1-4-0-eth-101/bridge vlan 101 default mcast 90 igmptimer 30
Bridge-path added successfully
```

3 Verify the bridge and bridge-path.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                       St  Table Data
-------------------------------------------------------------------------------
upl           Tagged  101    1/1/4/0/eth             1-1-4-0-eth-101/bridge        DWN S VLAN 101 default
1 Bridge Interfaces displayed

zSH> bridge-path show
VLAN/SLAN   Bridge                                Address
-------------------------------------------------------------------------------
       101 1-1-4-0-eth-101/bridge              Default, Age: 3600, MCAST Age: 90,
IGMP Query Interval: 30, Flap Mode: Default
```

Creating multicast control lists

Create a multicast control list, which defines which multicast addresses the remote-end video can access. Specifying a multicast control list of *0* allows all IP multicasts.

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 mcl 1
MCAST CONTROL LIST : 1
             224.1.1.1        224.1.1.2        224.1.1.3
```

Creating a downlink bridge on a GPON GEM port for video services

You can create a downlink bridge on a GPON GEM 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.

Specifying a multicast control list of *0* allows all IP multicasts. Members of the multicast control list must be defined to receive the video signal.

Create a downlink bridge with VLAN ID on an GPON GEM port.

A multicast control list entry of *0* allows all IP multicasts.

```
zSH> bridge add 1-1-2-1/gpononu gem 501 gtp 1 downlink vlan 101 tagged video 0/6
Adding bridge on 1-1-2-1/gponport
```

```
Created bridge-interface-record 1-1-2-501-gponport-101/bridge
```

Verify the bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------------
dwn             Tagged  101    1/1/2/1/gponou          1-1-2-501-gponport-101/bridge  DWN
upl             Tagged  101    1/1/4/0/eth             1-1-4-0-eth-101/bridge         DWN S VLAN 101 default
2 Bridge Interfaces displayed
```

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 1-1-2-501-gponport-101/bridge

bridge-interface-record  1-1-2-501-gponport-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: --------------------->  {0}
maxVideoStreams: ---------------------->  {5}
isPPPoA: ------------------------------>  {false}
floodUnknown: ------------------------->  {false}
floodMulticast: ----------------------->  {false}
bridgeIfEgressPacketRuleGroupIndex: -->  {0}
```

```
bridgeIfTableBasedFilter: ------------>  {NONE(0)}
bridgeIfDhcpLearn: ------------------->  {NONE(0)}
mvrVlan: ----------------------------->  {0}
vlan-xlate-from: --------------------->  {0}
slan-xlate-from: --------------------->  {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 GPON downlink bridge.

```
zSH> bridge delete 1-1-1-501-gponport-101/bridge
1-1-1-501-gponport-101/bridge delete complete
```

3 Delete the bridge path for the uplink bridge.

```
zSH> bridge-path delete 1-1-4-0-eth-101/bridge vlan 101 default
Delete complete
```

4 Delete the uplink bridge.

```
zSH> bridge delete 1-1-4-0-eth-101/bridge
1-1-4-0-eth-101/bridge delete complete
```

IGMP snooping with proxy reporting

This section describes:

- IGMP snooping overview, page 314
- Join and leave requests, page 315

- IGMP snooping with proxy reporting configuration (default IP address), page 316
- IGMPv3 messages respond for STBs, page 322

IGMP snooping overview

IGMP snooping applies to bridged video. Enabling IGMP snooping reduces traffic between the MXK-194/198 and the upstream multicast headend device by changing the behavior of the MXK-194/198 for more efficient tracking and grouping of JOIN and LEAVE requests. MXK-194/198 IGMP snooping also supports the following:

- Solicited or unsolicited query reports.
- Ability to configure the MXK-194/198 to send queries to hosts; by default the MXK-194/198 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 port 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 channels.
- MXK-194/198 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.

IGMP snooping with proxy reporting

This section describes:

- IGMP snooping with proxy reporting, page 314
- Join and leave requests, page 315
- IGMP snooping with proxy reporting configuration (default IP address), page 316
- IGMP snooping with proxy reporting with custom IP address, page 318
- Display bridge IGMP, page 321
- IGMPv3 messages respond for STBs, page 322

IGMP snooping with proxy reporting overview

IGMP snooping with proxy reporting applies to bridged video. IGMP snooping with proxy reporting reduces traffic between the MXK-194/198 and

the upstream multicast headend device by changing the behavior of the MXK-194/198 for more efficient tracking and grouping of JOIN and LEAVE requests. MXK-194/198 IGMP snooping with proxy reporting also supports the following:

- Solicited or unsolicited query reports.
- Ability to configure the MXK-194/198 to send queries to hosts; by default the MXK-194/198 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 port 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 channels.
- MXK-194/198 IGMP snooping with proxy reporting supports existing Max Video Streams and Multicast Control List functionality.
- Sending an IP address when a join request is sent to the upstream multicast headend device when proxy reporting is enabled.

Join and leave requests

For IGMP snooping without proxy reporting enabled, join requests from downstream hosts are simply forwarded by the MXK-194/198 to the multicast headend device. For IGMP snooping with proxy enabled, join requests from downstream hosts are not forwarded by the MXK-194/198 to the multicast headend device in the network, but are tracked by the MXK-194/198 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-194/198 terminates the join request from the host, originates new join request and sends it to the multicast headend device in the network along with the default IP address of 10.10.10.1 and a MAC address.

When a host sends a leave request that is the last leave request of the group, the MXK-194/198 terminates the leave request from the host and originates a new leave request and sends it to the multicast headend device in the network. 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-194/198.

In this way, the multicast headend device starts and stops video transmission by processing requests sent directly from the MXK-194/198 and not from downstream hosts. Proxy is when the MXK-194/198 sends join and leave requests to the network and snooping is when the MXK-194/198 monitors the join and leave requests from hosts to the MXK-194/198.

Figure 29: MXK-194/198 and multicast head end device join and leave requests and default IGMP IP address

IGMP snooping with proxy reporting configuration (default IP address)

After creating the uplink bridge, a new bridge path must be entered that will turn on proxy reporting for video on the uplink bridge by enabling IGMP snooping. Enabling proxy will send the default IP address 10.10.10.1.

Use the following syntax to enable proxy on the MXK-194/198:

```
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 and proxy reporting with default IP address

To enable IGMP snooping with proxy, enter **bridge-path-add** *interface/type* **vlan** *vlan-id* **igmpsnooping** *enable*:

1. Create a tagged uplink bridge on a FE/GE uplink and designate a VLAN ID.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 777 tagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-777/bridge
Bridge-path added successfully
```

The default for uplink bridges with VLAN IDs is tagged.

Verify the bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                      St  Table Data
-------------------------------------------------------------------------------------
upl            Tagged  777    1/1/4/0/eth             1-1-4-0-eth-777/bridge        DWN S VLAN 777 default
1 Bridge Interfaces displayed
```

2 Add the bridge path and enable IGMP snooping. The default is disable.

```
zSH> bridge-path add 1-1-1-0-eth-777/bridge vlan 777 default igmpsnooping enable
Bridge-path added successfully
```

Verify the bridge path.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
      777 1-1-4-0-eth-777/bridge            Default, Age: 3600, MCAST Age: 141, IGMP Query
Interval: 70, IGMP Proxy, Flap Mode: Default
```

Configuring IGMP snooping and proxy, multicast aging, and IGMP query

1 Create a tagged uplink bridge with VLAN ID. Uplink bridges on the MXK-194/198 default to tagged, even when *tagged* is not specified.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 777 tagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-777/bridge
Bridge-path added successfully
```

Verify the bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                      St  Table Data
-------------------------------------------------------------------------------------
upl            Tagged  777    1/1/4/0/eth             1-1-4-0-eth-777/bridge        DWN S VLAN 777 default
1 Bridge Interfaces displayed
```

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 **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 add 1-1-4-0-eth-777/bridge vlan 777 default igmpsnooping enable
mcast 120 igmptimer 60
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 1-1-4-0-eth-777/bridge            Default, Age: 3600, MCAST Age: 120, IGMP Query
Interval: 60, IGMP Proxy, Flap Mode: Default
```

Creating a downlink bridge on a GPON GEM port with video streams

You can create a downlink bridge on a GEM port using a VLAN ID and an SLAN ID. You can also specify a maximum number of video streams. 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. The default for the multicast control list is 0. Specifying a multicast control list of *0* allows all IP multicasts.

Create a downlink bridge on a GPON GEM port.

```
zSH> bridge add 1-1-1-1/gpononu gem 901 gtp 1 downlink vlan 777 video 0/3
Adding bridge on 1-1-1-1/gpononu
Created bridge-interface-record 1-1-1-901-gponport/bridge
```

Verify the bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                        St  Table Data
--------------------------------------------------------------------------------
upl            Tagged  777    1/1/4/0/eth               1-1-4-0-eth-777/bridge         DWN S VLAN 777 default
dwn                 777        1/1/1/1/gpononu          1-1-1-901-gponport/bridge      DWN
2 Bridge Interfaces displayed
```

IGMP snooping with proxy reporting with custom IP address

When IGMP snooping with proxy reporting is enabled on a static uplink bridge, the default source IP address in the Ethernet packet sent from the bridge is 10.10.10.0. In certain cases there may be a need to replace 10.10.10.1 with a custom Ethernet IP address. For example when a router in the network has implemented Reverse Path Forwarding and expects an IP address in the subnet of the router (see Figure 30). Another example is when different IP addresses in the same subnet are inserted for different SLMS devices for the purposes of debugging (see Figure 31).

Figure 30: MXK-194/198 with custom IGMP IP address

Figure 31: Multiple MXK-194/198s with custom IGMP IP addresses

Enabling IGMP snooping with proxy reporting and custom IP address

You can configure the MXK-194/198 to send a custom IP address used in proxy on the bridge path with the following syntax:

```
bridge-path add <interface/type> vlan <vlan-id> default igmpsnooping enable
igmpsendip enable <ipaddress>
```

1 Create the uplink bridge.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 777
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-777/bridge
Bridge-path added successfully
```

Verify the bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
upl            Tagged  777    1/1/4/0/eth              1-1-4-0-eth-777/bridge        DWN S VLAN 777 default
1 Bridge Interfaces displayed
```

2 Create the bridge path with **igmpsnooping** enabled and the custom IP address. Entering **igmpsendip** <*ipaddress*> provides the custom IP address for all IGMP messages going upstream from the MXK-194/198 to the network.

```
zSH> bridge-path add 1-1-4-0-eth-777/bridge vlan 777 default igmpsnooping enable
igmpsendip enable 172.16.1.3
Bridge-path added successfully
```

Verify the bridge path.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
      777 1-1-4-0-eth-777/bridge            Default, Age: 3600, MCAST Age: 141, IGMP Query
Interval: 70, IGMP Proxy, Custom IP 172.16.1.3, Flap Mode: Default
```

If you want to revert back to sending the default IP address of 10.10.10.1, enter this command:

```
zSH> bridge-path modify 1-1-4-0-eth-777/bridge vlan 777 default igmpsendip disable
```

3 If necessary, configure another MXK-194/198 for IGMP snooping with proxy reporting and a different custom IP address.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 555
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record 1-1-4-0-eth-555/bridge
Bridge-path added successfully
```

Verify the bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------------
upl            Tagged  555    1/1/4/0/eth              1-1-4-0-eth-555/bridge        DWN S VLAN 555 default
upl            Tagged  777    1/1/4/0/eth              1-1-4-0-eth-777/bridge        DWN S VLAN 777 default
2 Bridge Interfaces displayed
```

4 Create the bridge path with *igmpsnooping* enabled and the custom IP address. Entering *igmpsendip* <*ipaddress*> provides the IP address for all IGMP messages going upstream from the MXK-194/198 bridge to the network.

```
zSH> bridge-path add 1-1-4-0-eth-555/bridge vlan 555 default igmpsnooping enable
igmpsendip enable 172.16.1.4
Bridge-path added successfully
```

Verify the bridge path.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
--------------------------------------------------------------------------------
      777 1-1-4-0-eth-777/bridge            Default, Age: 3600, MCAST Age: 141, IGMP Query
Interval: 70, IGMP Proxy, Custom IP 172.16.1.3, Flap Mode: Default
      555 1-1-4-0-eth-555/bridge            Default, Age: 3600, MCAST Age: 141, IGMP Query
Interval: 70, IGMP Proxy, Custom IP 172.16.1.4, Flap Mode: Default
```

If you want to revert back to sending the default IP address of 10.10.10.1, enter this command:

```
zSH> bridge-path modify 1-1-4-0-eth-555/bridge vlan 555 default igmpsendip disable
```

5 Create a downlink bridge for video on a GPON GEM port.

```
zSH> bridge add 1-1-1-1/gpononu gem 901 gtp 1 downlink vlan 555 tagged video
0/3
Adding bridge on 1-1-1-1/gpononu
Created bridge-interface-record 1-1-1-901-gponport-555/bridge
```

Deleting IGMP snooping with proxy and custom IP

Delete the uplink bridge.

```
zSH> bridge delete 1-1-4-0-eth-555/bridge vlan 555
Bridge-path deleted successfully
1-1-4-0-eth-555/bridge delete complete
```

Display bridge IGMP

Displaying bridge IGMP

The **bridge igmp** command displays the time left for multicast.

1 To view the time left on a multicast feed enter:

```
zSH> bridge igmp
Slan Vlan       MAC Address        MCAST IP                  Bridge Host MAC   LastJoinTimer
--------------------------------------------------------------------------------
   0  999 01:00:5e:0a:0a:0a    224.10.10.10             1-1-6-0-eth 00:10:a4:c4:fe:95 02:14
   0  999 01:00:5e:7f:ff:fa 239.255.255.250              1-1-6-0-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.

IGMPv3 messages respond for STBs

Some newer versions of Set Top Box (STB) support IGMPv2 and IGMPv3. This MXK-194/198 release does not support IGMPv3 but it can respond to the IGMPv3 messages. When the MXK-194/198 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

View IGMP bridge statistics.

```
zSH> bridge igmpstat
                                          Received                                Transmitted
Interface                          GenQuery SpecQuery v2Report   Leave      GenQuery SpecQuery v2Report   Leave
1-1-1-501-gponport-510/bridge            627        0        0        0             0        0        0        0
1-1-2-0-eth-105/bridge                     0        0        0        0             0        0        0        0
2 Bridge Interfaces displayed
```

10 ETHERNET SERVICES

This chapter explains the Ethernet services on the MXK-194/198.

The FE/GE interfaces are identified as **1-1-x-0/eth** in the command line, where x is 2 to 9. The 10GE uplink interfaces are identified as **1-1-x-0/eth** in the command line, where x is 10 or 11.

Bridging with linear 10 Gigabit Ethernet

Within the linear topology, bridging can be configured to forward traffic based on MAC address and VLAN ID to an IP or outside network. The node connected to the network contains a bridge uplink and bridge path on the MXK-194/198 first 10 Gigabit Ethernet port (*1-1-10-0/eth*) to direct all bridged traffic to the outside or IP network. This device also contains a intralink on the second 10 Gigabit Ethernet port (*1-1-11-0/eth*) so unknown traffic is sent to the downstream, even though address learning is not enabled.

The second node in the daisy-chained linear topology contains a bridge uplink on the first 10 Gigabit Ethernet port (*1-1-10-0/eth*) to direct all outgoing bridged traffic to the upstream node. This node also contains a intralink bridge on the second 10 Gigabit Ethernet port (*1-1-11-0/eth*) so unknown traffic is sent to the downstream to another network or subtended Ethernet device, even though address learning is not enabled.

Additional MXK-194/198 nodes can be added to the daisy-chained linear topology by repeating this pattern of connections and bridging.

Figure 32 illustrates the bridging using the MXK-194/198 10 Gigabit Ethernet linear configuration.

Figure 32: Bridging using the 10 Gigabit Ethernet linear configuration

Configuring bridging

1 On the node connected to the Ethernet or IP network

a Add a bridge interface to the first 10 GE uplink port (this is the port connected to the external network):

```
zSH> bridge add 1-1-10-0/eth uplink vlan 555
Adding bridge on 1-1-10-0/eth
Created bridge-interface-record 1-1-10-0-eth-555/bridge
Bridge-path added successfully
```

b Verify the uplink bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------------------------
upl             Tagged  555    1/1/10/eth              1-1-10-0-eth-555/bridge       DWN S VLAN 555 default
```

```
1 Bridge Interfaces displayed
```

All bridge traffic will be forwarded over this interface.

c Add an intralink bridge on the second 10 GE port:

```
zSH> bridge add 1-1-11-0/eth intralink vlan 555
Adding bridge on 1-1-11-0/eth
Created bridge-interface-record 1-1-11-0-eth-555/bridge
Bridge-path added successfully
```

Unlearned traffic received on this interface is forwarded to the external network.

This interface is the intralink for the node.

d Add a downlink bridge to the remote subscriber:

```
zSH> bridge add 1-1-1-1/gpononu gem 501 gtp 1 downlink vlan 555
Adding bridge on 1-1-1-1/gpononu
Created bridge-interface-record 1-1-1-501-gponport/bridge
```

e Verify the bridge interfaces.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                    St  Table Data
-------------------------------------------------------------------------------
dwn                555      1/1/1/1/gpononu          1-1-1-901-gponport/bridge     UP
int            Tagged 555   1/1/11/0/eth             1-1-11-0-eth-555/bridge        DWN S VLAN 555 Intralink
upl            Tagged 555   1/1/10/0/eth             1-1-10-0-eth-555/bridge       DWN S VLAN 555 default
3 Bridge Interfaces displayed
```

2 On the next node in the linear daisy-chain configuration:

a Add an uplink bridged interface:

```
zSH> bridge add 1-1-10-0/eth uplink vlan 555
Adding bridge on 1-1-10-0/eth
Created bridge-interface-record 1-1-10-0-eth-555/bridge
Bridge-path added successfully
```

b Add an intralink bridge on the second 10 GE port:

```
zSH> bridge add 1-1-11-0/eth intralink vlan 555
Adding bridge on 1-1-11-0/eth
Created bridge-interface-record 1-1-11-0-eth-555/bridge
Bridge-path added successfully
```

Unlearned traffic received on this interface is forwarded to the external network.

This interface is the intralink for the node.

c Add a downlink bridge to the remote subscriber:

```
zSH> bridge add 1-1-2-1/gpononu gem 901 gtp 1 downlink vlan 555
Adding bridge on 1-1-2-1/gpononu
Created bridge-interface-record 1-1-2-901-gponport/bridge
```

d View the bridge interfaces.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------
dwn                555      1/1/2/1/gpononu         1-1-2-901-gponport/bridge           UP
int            Tagged  555    1/1/11/0/eth              1-1-11-0-eth-555/bridge        DWN S VLAN 555 Intralink
upl            Tagged  555    1/1/10/0/eth               1-1-10-0-eth-555/bridge        DWN S VLAN 555 default
3 Bridge Interfaces displayed
```

3 On the next node in the linear daisy-chain configuration:

a Add a bridge interface to the first 10 GE uplink port (this is the port connected to the second node):

```
zSH> bridge add 1-1-10-0/eth uplink vlan 555
Adding bridge on 1-1-10-0/eth
Created bridge-interface-record 1-1-10-0-eth-555/bridge
Bridge-path added successfully
```

b Verify the uplink bridge.

All bridge traffic will be forwarded over this interface.

```
zSH> bridge show
Type VLAN        Bridge                          St  Table Data
-------------------------------------------------------------------------------
upl Tagged 555   1-1-10-0-eth-555/bridge         DWN S VLAN 555 default
```

c Add an intralink bridge on the second 10 GE port:

```
zSH> bridge add 1-1-11-0/eth intralink vlan 555
Adding bridge on 1-1-11-0/eth
Created bridge-interface-record 1-1-11-0-eth-555/bridge
Bridge-path added successfully
```

Unlearned traffic received on this interface is forwarded to the external network.

This interface is the intralink for the node.

d Add a downlink to the remote subscriber:

```
zSH> bridge add 1-1-3-2/gpononu gem 456 gtp 1 downlink vlan 555
Adding bridge on 1-1-3-2/gpononu
Created bridge-interface-record 1-1-3-456-gponport/bridge
```

e Verify the bridge interfaces.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                Bridge                        St  Table Data
-------------------------------------------------------------------------------
dwn                555        1/1/3/2/gpononu        1-1-3-456-gponport/bridge           UP
int            Tagged  555    1/1/11/0/eth              1-1-11-0-eth-555/bridge        DWN S VLAN 555 Intralink
upl            Tagged  555    1/1/10/0/eth              1-1-10-0-eth-555/bridge        DWN S VLAN 555 default
3 Bridge Interfaces displayed
```

4 Continue this configuration for all the nodes in the daisy-chain connection.

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

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 1-1-3-0/eth
Interface 1-1-3-0/eth
        Physical location:     1/1/3/0/eth

        Administrative status: up
        Port type specific information:
                Link state mirroring not configured.
```

Use port up, down, or bounce to alter the administrative status of a physical or virtual interface. Bounce performs a down operation followed by an up operation.

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

```
zSH> port up 1-1-3-0/eth
1-1-3-0/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 1-1-3-0/eth
1-1-3-0/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 1-1-3-0/eth
1-1-3-0/eth set to admin state DOWN
1-1-3-0/eth set to admin state UP
```

Enter the **port status** <*interface*> to get the operational status, speed and duplex mode of the Ethernet port.

```
zSH> port status 1-1-1-0/eth
Operational status  : Up
Rate in Bps         : 100000000
Duplex              : Full
```

11 GPON Subscriber Interfaces

This chapter explains how to configure GPON connections on the MXK-194/198. It contains the following sections:

- Overview, page 329
- Smart OMCI GPON zNID installation, page 329
- Dynamic OMCI GPON zNID installation, page 362
- Browser based GPON zNID installation, page 418
- GPON type B redundancy, page 436
- MXK-194/198 GPON using the Reg ID for provisioning, page 443
- GPON extended reach, page 447
- Received Signal Strength Indication (RSSI) and Digital Diagnostic Monitoring (DDM), page 450
- Manage ONU with OMCI, page 452
- Bandwidth allocation on T-cont, page 471
- GEM port creation, page 484
- OMCI Statistics, page 491
- GPON Alarms, page 494

Overview

MXK-194/198 GPON interfaces provide a standards-based, high-speed GPON interface between the MXK-194/198 and CPE devices.

This chapter provides the step-by-step configuration procedures to configure Data, Voice, and Video services on MXK-194/198 for Smart OMCI based ONTs, Dynamic based ONTs, or Browser based ONTs.

Smart OMCI GPON zNID installation

This section includes the following topics:

- OMCI overview, page 331

- Smart OMCI overview, page 331
- OMCI GPON zNID installation with Smart OMCI, page 334
- Delete the OMCI profile, page 353
- Import and export the OMCI profile, page 357

Figure 33: Installation procedure for OMCI GPON zNID with Smart OMCI

OMCI overview

The ONT Management and Control Interface (OMCI) is a protocol defined by ITU-T G.988 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 ONU management and control interface requirements given in the ITU-T G.988. Recommendation are needed to manage ONU in the following areas:

- Configuration management
- Fault management
- Performance management
- Security management

Smart OMCI overview

OMCI Profiles

Smart OMCI functionality is implemented on the MXK-194/198 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 Figure 34, these three profile types have a hierarchical relationship.

Figure 34: Smart OMCI Architecture

- ME profile

  The ME profile defines an ONU model and service profile.

  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-194/198. The ME profile is created on the MXK-194/198 by an ME profile file that is downloaded from Zhone's website.

- Generic profile

  The Generic profile defines the common default parameters for 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-194/198 (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 give values to parameters per user based before activating the end-user. The Specific profile is always associated with only one Generic profile. The Specific profile contains value for specific users, 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 be 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.

Dynamic GEM ports

If an ONU is planned to be managed with Smart OMCI, when you create a GEM port, make sure the GEM port ID = GEM index + ONU ID, where GEM index is the port offset selected in the Smart OMCI web interface, from 5xx to 35xx.

Each ONU supports up to 16 GPON GEM ports.

Figure 35: Dynamic GEM port ID are created from the GEM index and the ONU ID

When creating downstream services on the MXK-194/198, the subport information in the **bridge add** command would be the same as the GEM port ID.

```
zSH> bridge add 1-1-4-542/gponport gtp 1 downlink vlan 200 tagged
Adding bridge on 1-1-4-542/gponport
Created bridge-interface-record 1-1-4-542-gponport-200/bridge
```

In the above example, GEM port 1-1-4-542 has been created on ONU 1-1-4-42/gpononu. The GEM port ID, 542, is the sub-port for the **bridge add** command, and it is in the **bridge add** command which defines which VLAN is matched to the GEM port.

Figure 36: zNID 1 and 42 are from the same company. zNID 2 and 3 are from separate residences

OMCI GPON zNID installation with Smart OMCI

Generally these are the steps to follow to configure the MXK-194/198 to be able to manage OMCI GPON zNID with Smart OMCI:

- Create a ME profile through SMART OMCI web-interface, page 335
- Download a ME profile file to the MXK-194/198, page 339
- Create a ME profile for the selected ONT model, page 340
- Create Generic profiles for service plan, page 340

- Create high speed Internet on GPON OMCI on uplink and downlink bridges, page 343
- Create an uplink and downlink bridge on GPON OMCI for video, page 348
- Create a TLS bridge on GPON OMCI for VoIP, page 351

Create a ME profile through SMART OMCI web-interface

Zhone Technologies provides the service provider a Smart OMCI web-interface to select desired ONU model and services.

Creating ME profile file through Smart OMCI web-interface

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 to 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* on page 484 for configuration information.

6 Click the **Create Configuration File** button. An ME profile file is created and displayed in the ME profile file page.

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).

Download a ME profile file to the MXK-194/198

Downloading ME profile file to MXK-194/198

The ME profile file must be downloaded from a TFTP/ SFTP server to MXK-194/198.

1 Verify that the current directory is the root (i.e. /card1) with the **pwd** command. If not in this directory use the **cd** (change directory) command to move to it.

```
zSH> pwd

/card1
```

2 Create a directory at the root level (i.e. /card1), 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
```

3 Move to the newly created directory.

```
zSH> cd me
```

4 Download the ME profile file to the current directory in the MXK-194/198 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-194/198 */me* directory, and save the ME profile file with the same name.

```
zSH> file download 172.16.80.201 /
ZNID-GPON-2510-omci.txt ZNID-GPON-2510-omci.txt
Bytes copied:     18411
File download successful
```

Create a ME profile for the selected ONT 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.

Creating ME profile for selected ONT model

Create an ME profile from the ME profile file. One ME profile is created for each ONU model.

1 Create an ME profile. This example creates a ME profile from the downloaded ME profile file ZNID-GPON-2510-omci.txt, and name it to 2510-tripleplay-me.

```
zSH> gpononu profile create me 2510-tripleplay-me
ZNID-GPON-2510-omci.txt
Profile created
```

2 Verify the created ME profile name:

```
zSH> gpononu profile show me
2510-tripleplay-me
```

Create Generic profiles for service plan

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.

Creating Generic profiles for service plan

To create a Generic profile:

1. Create a Generic profile:

```
zSH> gpononu profile create gen 2510-tripleplay-gen
2510-tripleplay-me
Profile created
```

2. Verify the created Generic profile name.

```
zSH> gpononu profile show gen
2510-tripleplay-gen
```

3. Update the Generic profile.

   To assign or change a parameter, enter the line number, click Enter, then enter the value, at last enter s to save the profile.

```
zSH> gpononu profile update gen 2510-tripleplay-gen
Generic Profile: 2510-tripleplay-gen
1  "ETH1 Auto Detection [0]"
2  "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"
3  "ETH 1 Data VLAN 2 (VID or COS,VID) [0,100]"
4  "ETH2 Auto Detection [0]"
5  "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"
6  "ETH 2 Data VLAN 2 (VID or COS,VID) [0,100]"
7  "ETH3 Auto Detection [0]"
8  "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
9  "ETH 3 Video VLAN 2 (VID or COS,VID) [4,999]"
10  "ETH4 Auto Detection [0]"
11  "Voice VLAN [7,200]"
12  "VOIP Host IP Option: 2-static, 3-DHCP [2]"
13  "VOIP Host IP [0.0.0.0]"
14  "VOIP Netmask [0.0.0.0]"
15  "VOIP Gateway [0.0.0.0]"
16  "VOIP Server IP [0.0.0.0]"
17  "VOIP Primary DNS [0.0.0.0]"
18  "VOIP Secondary DNS [0.0.0.0]"
19  "Country Code [ 1]"
20  "Rx Gain [0]"
21  "Tx Gain [0]"
22  "Out-of-band DTMF [0]"
23  "Echo Cancel: 1-enable, 0-disable [1]"
24  "POTS1 Dial Number [1111]"
25  "POTS1 User Name [11111]"
26  "POTS1 Password [11111]"
27  "POTS2 Dial Number [2222]"
28  "POTS2 User Name [22222]"
29  "POTS2 Password [22222]"
```

```
30  "Fax Mode [0]"
31  "CID Features [63]"
32  "Call Waiting Features [3]"
33  "Call Progress or Transfer Features [255]"
34  "Call Present Features [15]"
35  "ETH 1 Admin Status: 0-Up, 1-Down [0]"
36  "ETH 2 Admin Status: 0-Up, 1-Down [0]"
37  "ETH 3 Admin Status: 0-Up, 1-Down [0]"
38  "ETH 4 Admin Status: 0-Up, 1-Down [0]"
39  "POTS 1 Admin Status: 0-Up, 1-Down [0]"
40  "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:2
"ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]" : 100
Enter OMCI edit command: 3
"ETH 1 Data VLAN 2 (VID or COS,VID) [0,100]" : 100
...
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 profile update gen** command and enter OMCI edit command **L** (not case sensitive).

```
zSH> gpononu profile update gen 2510-tripleplay-gen
Generic Profile: 2510-tripleplay-gen
1  "ETH1 Auto Detection [0]"
2  "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"  100
3  "ETH 1 Data VLAN 2 (VID or COS,VID) [0,100]"  100
4  "ETH2 Auto Detection [0]"
5  "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"
6  "ETH 2 Data VLAN 2 (VID or COS,VID) [0,100]"
7  "ETH3 Auto Detection [0]"
8  "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
9  "ETH 3 Video VLAN 2 (VID or COS,VID) [4,999]"
10  "ETH4 Auto Detection [0]"
11  "Voice VLAN [7,200]"
12  "VOIP Host IP Option: 2-static, 3-DHCP [2]"
13  "VOIP Host IP [0.0.0.0]"
14  "VOIP Netmask [0.0.0.0]"
15  "VOIP Gateway [0.0.0.0]"
16  "VOIP Server IP [0.0.0.0]"
17  "VOIP Primary DNS [0.0.0.0]"
18  "VOIP Secondary DNS [0.0.0.0]"
19  "Country Code [ 1]"
```

```
20  "Rx Gain [0]"
21  "Tx Gain [0]"
22  "Out-of-band DTMF [0]"
23  "Echo Cancel: 1-enable, 0-disable [1]"
24  "POTS1 Dial Number [1111]"
25  "POTS1 User Name [11111]"
26  "POTS1 Password [11111]"
27  "POTS2 Dial Number [2222]"
28  "POTS2 User Name [22222]"
29  "POTS2 Password [22222]"
30  "Fax Mode [0]"
31  "CID Features [63]"
32  "Call Waiting Features [3]"
33  "Call Progress or Transfer Features [255]"
34  "Call Present Features [15]"
35  "ETH 1 Admin Status: 0-Up, 1-Down [0]"
36  "ETH 2 Admin Status: 0-Up, 1-Down [0]"
37  "ETH 3 Admin Status: 0-Up, 1-Down [0]"
38  "ETH 4 Admin Status: 0-Up, 1-Down [0]"
39  "POTS 1 Admin Status: 0-Up, 1-Down [0]"
40  "POTS 2 Admin Status: 0-Up, 1-Down [0]"
Enter OMCI edit command or [s]ave, [q]uit, [h]elp:l

ID    Generic Profile: 2510-tripleplay-gen
====
=========================================================
==================
   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    : 100
      Default Value: 0,100
   3  Name         : $autoDetectConfigEth2
      Comment      : ETH2 Auto Detection
      Type         : string(32)
      Gen Value    : 100
      Default Value: 0,100
<SPACE> for next page, <CR> for next line, A for all, Q
to quitq
```

Create high speed Internet on GPON OMCI on uplink and downlink bridges

The high speed Internet application uses uplink and downlink bridges with a VLAN ID. You should notice from the flowchart and procedures that provisioning video also uses uplink/downlink bridge configurations, just the GEM port setup (from the OMCI profile), GPON traffic profile and VLAN are different. For triple play services (As long as the OMCI profiles are

configured properly) you can add the video bridge or VoIP bridge in the same process. For ease of discussion each of the applications is described separately in this chapter.

For data service we will create uplink/downlink bridges with VLAN 100.

Creating the GPON traffic profile

GPON traffic profiles are a template for defining how traffic will be handled on the bridge with which the GTP is associated. One GTP may be associated with many different bridges. The GTP in this procedure will create a high bandwidth configuration.

Refer to *Create a GEM port* on page 484 and *Configure GPON traffic profile* on page 472 to get detail configuration and parameter description.

The following is recommended for high speed data configurations.

```
zSH> new gpon-traffic-profile 1
gpon-traffic-profile  1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 1024
traffic-class: ---------->  {ubr}:     ubr is the default value
compensated: ------------>  {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.
```

Creating uplink and downlink bridges

Create an uplink and downlink bridge for VLAN 100:

1 Create the uplink bridge interface:

a Add the bridge interface for the uplink.

Make sure VLAN ID matches the VLAN ID you assigned for data service in the Generic Profile. This example, data services uses VLAN 100.

```
zSH> bridge add 1-1-5-0/eth uplink vlan 100
Adding bridge on 1-1-5-0/eth
Created bridge-interface-record ethernet5/bridge
Bridge-path added successfully
```

2 Create downlink bridge interface:

Uses the GEM index assigned in the Smart OMCI web tool to calculate the GEM port ID with the following formula:

GEM port ID = GEM index + ONU ID

This example uses GEM index 5xx for data service, and ONT ID is 1/4/**1**, so the GEM port ID is 501.

```
zSH> bridge add 1-1-4-501/gponport gtp 1 downlink vlan 100 tagged
Adding bridge on 1-1-4-501/gponport
Created bridge-interface-record 1-1-4-501-gponport-100/bridge
```

Creating the Specific profile for a new user

On MXK-194/198 create and modify Specific profile for each user; in the case of specific profiles, the OMCI supports are associated with the ONT.

Only one Specific profile can be added on an ONT.

To add a new user:

1 Create and modify the Specific profile.

a Create the Specific profile, selecting the ME profile and Generic profile to associate with the Specific (user) profile.

```
zSH> gpononu profile create spec 1/4/1
2510-tripleplay-me 2510-tripleplay-gen
Profile created
```

b Update the Specific profile.

```
zSH> gpononu profile update spec 1/4/1
Specific Profile: 1/4/1
   1  "ETH1 Auto Detection [0]"
   2  "ETH 1 Data VLAN 1 (VID or COS,VID) [0,100]"  100
   3  "ETH 1 Data VLAN 2 (VID or COS,VID) [0,100]"  100
   4  "ETH2 Auto Detection [0]"
   5  "ETH 2 Data VLAN 1 (VID or COS,VID) [0,100]"
   6  "ETH 2 Data VLAN 2 (VID or COS,VID) [0,100]"
   7  "ETH3 Auto Detection [0]"
   8  "ETH 3 Video VLAN 1 (VID or COS,VID) [4,999]"
   9  "ETH 3 Video VLAN 2 (VID or COS,VID) [4,999]"
   10  "ETH4 Auto Detection [0]"
   11  "Voice VLAN [7, 200]"
   12  "VOIP Host IP Option: 2-static, 3-DHCP [2]"
   13  "VOIP Host IP [0.0.0.0]"
   14  "VOIP Netmask [0.0.0.0]"
   15  "VOIP Gateway [0.0.0.0]"
   16  "VOIP Server IP [0.0.0.0]"
   17  "VOIP Primary DNS [0.0.0.0]"
   18  "VOIP Secondary DNS [0.0.0.0]"
   19  "Country Code [ 1]"
   20  "Rx Gain [0]"
   21  "Tx Gain [0]"
   22  "Out-of-band DTMF [0]"
   23  "Echo Cancel: 1-enable, 0-disable [1]"
   24  "POTS1 Dial Number [1111]"
```

```
   25  "POTS1 User Name [11111]"
   26  "POTS1 Password [11111]"
   27  "POTS2 Dial Number [2222]"
   28  "POTS2 User Name [22222]"
   29  "POTS2 Password [22222]"
   30  "Fax Mode [0]"
   31  "CID Features [63]"
   32  "Call Waiting Features [3]"
   33  "Call Progress or Transfer Features [255]"
   34  "Call Present Features [15]"
   35  "ETH 1 Admin Status: 0-Up, 1-Down [0]"
   36  "ETH 2 Admin Status: 0-Up, 1-Down [0]"
   37  "ETH 3 Admin Status: 0-Up, 1-Down [0]"
   38  "ETH 4 Admin Status: 0-Up, 1-Down [0]"
   39  "POTS 1 Admin Status: 0-Up, 1-Down [0]"
   40  "POTS 2 Admin Status: 0-Up, 1-Down [0]"
   Enter OMCI edit command: 24
   "POTS1 Dial Number [1111]" : 2012000984
   Enter OMCI edit command: 25
   "POTS1 User Name [11111]" : 2012000984
   Enter OMCI edit command: 26
   "POTS1 Password [11111]": password
   ...
   Enter OMCI edit command: s
SPECIFIC profile has been saved
```

2 Make sure every variable has an assigned value.

To view the current settings of configuration variables on ONU 1/4/1 enter **gpononu profile show vars 1/4/1** command.

```
zSH> gpononu profile show vars 1/4/1
              Variable Description                      Value          Source
    ------------------------------------------  ------------------  --------
  1 ETH1 Auto Detection                          0 (Auto-sensing)    Default
  2 ETH2 Auto Detection                          0 (Auto-sensing)    Default
  3 ETH3 Auto Detection                          0 (Auto-sensing)    Default
  4 ETH4 Auto Detection                          0 (Auto-sensing)    Default
  5 T1/E1 1 Admin Status: 0-Up, 1-Down           1 (down)            Default
  6 T1/E1 2 Admin Status: 0-Up, 1-Down           1 (down)            Default
  7 T1/E1 3 Admin Status: 0-Up, 1-Down           1 (down)            Default
  8 T1/E1 4 Admin Status: 0-Up, 1-Down           1 (down)            Default
  9 ETH 1 Admin Status: 0-Up, 1-Down             0 (up)              Default
 10 ETH 2 Admin Status: 0-Up, 1-Down             0 (up)              Default
 11 ETH 3 Admin Status: 0-Up, 1-Down             0 (up)              Default
 12 ETH 4 Admin Status: 0-Up, 1-Down             0 (up)              Default
 13 POTS 1 Admin Status: 0-Up, 1-Down            1 (down)            Default
 14 POTS 2 Admin Status: 0-Up, 1-Down            1 (down)            Default
```

Activating ONT

Activate the ONT to add it to the system. If you are adding multiple services, you would range the ONT after all the services have been added.

Note: Only run the **gpononu set** command once to add the ONT. If the ONT has been activated and the OMCI profiles are configured for other service, you may add other bridges without resetting the ONT. If you change OMCI profiles you will need to resync/reboot the ONT. To resync ONT use the **gpononu resync <slot>[/<olt>[/<onu>]]** command. To reboot ONT use the **gpononu reboot <slot>[/<olt>[/<onu>]]** command.

1. To activate an ONT first run the **gpononu show** command to display the ONTs currently on the OLT, and discover the available serial numbers.

   The **gpononu show** command has options to select by slot and OLT. If you run the command without defining the slot/OLT the command will check for ONTs on every port of every card and depending on the number of cards, may take a long time to complete.

```
zSH> gpononu show 1/4
Processing list of 128
This command may take several minutes to complete.
Do you want to continue?  (yes or no) [no] yes
Free ONUs for slot 1 olt 4:
    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 1 olt 4:
sernoID   Vendor  Serial Number     sernoID   Vendor  Serial Number
    1      CIGG    138543368
```

2. Run the **gpononu set** command to associate a sernoID to the appropriate ONT. This example associates sernoID 1 to ONU 1/4/1:

```
zSH> gpononu set 1/4/1 1
Onu 1 successfully enabled with serial number CIGG
138543368
```

3. Run the **gpononu show** command to verify the ONT is enabled, and OMCI support is added into the ONT (the associated ME profile and Generic profile can be displayed).

```
zSH> gpononu show 1/4/1
                                        Serial
ONU       Name        Enabled Model # Number         OMCI files and profiles
=== ================= ======= ======= ============== ========================
  1           1-1-4-1   Yes    2510   CIGG 138543368 ME   2510-config1
                                                     GEN  2510-service-plan1
```

4. Run the **gpononu status** command to verify the OMCI Config State is active.

```
zSH> gpononu status 1/4/1
```

```
                          Omci       Gpon    Download    OLT      ONT     Distance
ID    Onu   OperStatus ConfigState OnuStatus  State    Rx Power  Rx Power   (KM)
== ======== ========== =========== ========= ========= ========= ========= =======
1  1-1-4-1     Up     Active      Active    NoUpgrade -23.8 dBm -23.0 dBm   18
```

5 Run the **bridge show** command to view the MAC address of the connected PC.

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN Physical        Bridge                       St Table Data
-------------------------------------------------------------------------------
upl            Tagged 100 1/1/5/0/eth    ethernet5-100/bridge        UP S VLAN 100 default
dwn            Tagged 100 1/1/4/1/gpononu 1-4-4-501-gponport-100/bridge
UP D 00:00:86:43:3c:e4 MAC of PC
```

Testing the data bridge

The steps described below are specific to using a PC where a DHCP server is either present on the MXK or across a bridge with bridgeIfCustomDhcp enabled on the MXK.

Verify that the user can get data on the PC:

1 Connect an ONT downlink ethernet port to the PC.

   Make sure the ONT model matches the one you assigned with the Smart OMCI web tool. This example connects a ZNID-GPON-2510 to the PC.

   And also make sure the ONT downlink ethernet port number matches the one you assigned with the Smart OMCI web tool for data service. In this example, you can connect either ETH 1 or ETH 2 to the PC.

2 Open a command prompt on the PC and enter **ipconfig** to verify that you can get an IP address from DHCP server for the PC.

3 Open an internet browser on the PC, you should be able to access the internet now.

Create an uplink and downlink bridge on GPON OMCI for video

Video bridging is similar to data bridging and uses downlink/uplink bridges, however, the GPON traffic profile, GEM ports and VLANs are different.

Creating GPON traffic profile

Add the GPON traffic profile.

The following GPON traffic profile is recommended for video:

```
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}:
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.
```

Creating an uplink and downlink bridge

Create an uplink and downlink bridge for VLAN 999:

1 Create an uplink bridge interface

a Create the uplink bridge interface

The following example creates a video uplink bridge interface with proxy reporting and 30 second igmp query interval.

```
zSH> bridge add 1-1-5-0/eth uplink vlan 999 tagged

Adding bridge on 1-1-5-0/eth
Created bridge-interface-record ethernet5-999/
bridge
Bridge-path added successfully
```

b Modify the **bridge-path** for the uplink. Note how the *igmptimer* is added to the bridge-path.

```
zSH> bridge-path modify ethernet5-999/bridge vlan
999 default igmpsnooping enable igmptimer 30
```

2 Create downlink bridge interface.

Create a downlink bridge on a GPON port with VLAN ID and GPON traffic profile.

You can also specify option **video** *m*/*n*. *m* indicates the multicast control list, *n* indicates the maximum video streams. By specifying **video 0/4** in this example you can enable subscriptions up to four video streams on the interface without control list checking.

If you want to have multicast control list checking, use the **new mcast-control-entry** command to create a multicast control list first.

```
zSH> bridge add 1-1-4-901/gponport gtp 2 downlink vlan
999 tagged video 0/4
Adding bridge on 1-1-4-901/gponport
Created bridge-interface-record
1-1-4-901-gponport-999/bridge
```

3 Enter the **bridge show** command to view the MAC address of the connected PC.

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN Physical         Bridge                        St Table Data
-------------------------------------------------------------------------------
upl           Tagged 100 1/1/5/0/eth   ethernet5-100/bridge           UP  S VLAN 100
default
dwn           Tagged 100 1/4/4/1/gpononu 1-4-4-501-gponport-100/bridge  UP  D
00:00:86:43:3c:e4
upl           Tagged 999  1/1/5/0/eth   ethernet5-999/bridge           UP  S VLAN 999
default
dwn          Tagged 999  1/1/4/1/gpononu 1-4-4-901-gponport-999/bridge  UP  D
00:00:87:44:0c:e7  MAC of PC
                                                                            D
01:00:5e:0a:0a:0a
```

Because Specific profile was already created on this ONT when configuring the data application, you do not need to create a Specific profile again.

Since you only add the ONT once, you would normally run the **gpononu set** command after you have added all the services. You may add service after activating the ONT, however if you change the OMCI profiles later, you need to resync or reboot the ONT. See the Step 1 *Activate the ONT* in the data application for the command and greater detail on the operation.

Testing the IPTV bridge

When using a PC and software to emulate a set top box (STB), use **ping** to verify that the video server is alive.

1 Connect an ONT downlink ethernet port to the customer video equipment. This example connects to a PC that runs a STB emulation software.

   Make sure the ethernet port number matches the one you assigned with the Smart OMCI web tool for video service. In this example you can connect either ETH 3 or ETH 4 to the PC.

2 Open a command prompt on the PC and enter **ipconfig** to verify that you can get an IP address for the PC.

3 Ping the video server

   a Open a DOS window

   b Ping the upstream gateway (provided in your environment setup)

4 Open the STB emulation software and connect to the video server.

   As long as you can ping you are showing that you have a data path through the zNID and the MXK-194/198 to the video server. You should be able to connect to the video stream with the STB emulation software.

Create a TLS bridge on GPON OMCI for VoIP

For VoIP service we recommend using TLS bridging.

Creating GPON traffic profile

Add the GPON traffic profile.

The following GPON traffic profile is recommended for up to four VoIP phones or four POTS ports.

```
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}:
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..
```

Creating a TLS bridge

Create a TLS bridge for VLAN 300:

1. Create a TLS bridge on the uplink interface.

```
zSH> bridge add 1-1-5-0/eth tls vlan 300 tagged
Adding bridge on 1-1-5-0/eth
Created bridge-interface-record ethernet5-300/bridge
Bridge-path added successfully
```

2. Create a TLS bridge on the downlink interface.

```
zSH> bridge add 1-1-4-701/gponport gtp 3 tls vlan 300
tagged
Adding bridge on 1-1-4-701/gponport
Created bridge-interface-record
1-1-4-701-gponport-300/bridge
Bridge-path added successfully
```

3. On MXK-194/198, run the **bridge show** command to view the MAC address of the connected VoIP phone.

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN  Physical        Bridge                        St Table Data
```

```
-------------------------------------------------------------------------------
upl            Tagged 100  1/1/5/0/eth      ethernet5-100/bridge           UP  S VLAN 100
default
dwn            Tagged 100  1/1/4/1/gpononu  1-1-4-501-gponport-100/bridge  UP  D
00:00:86:43:3c:e4
upl            Tagged 999  1/1/5/0/eth      ethernet5-999/bridge           UP  S VLAN 999
default
dwn            Tagged 999  1/1/4/1/gpononu  1-1-4-901-gponport-999/bridge  UP  D
00:00:87:44:0c:e7
                                                                         D 01:00:5e:0a:0a:0a
tls            Tagged 300  1/1/4/1/gpononu  1-1-4-701-gponport-300/bridge  UP  D
00:19:c7:02:9c:6b MAC of Phone
tls            Tagged 300  1/1/5/0/eth      ethernet5-300/bridge           UP  D
00:00:86:43:3c:e4
                                                                         D 00:00:86:43:ec:69
                                                                         D 00:01:47:1a:e4:74
                                                                         D 00:03:e3:97:bb:00
                                                                         D 00:50:04:78:56:85
                                                                         D 00:50:04:bf:63:3e
```

Because a Specific profile is already created on this ONT when configuring data application, you do not need to create a Specific profile again.

Since you only add the ONT once, you would normally run the **gpononu set** command after you have added all the services. You may add service after activating the ONT, however if you change the OMCI profiles later, you need to resync or reboot the ONT. See the Step 1 *Activate the ONT* in the data application for the command and greater detail on the operation.

Testing the VoIP configuration

1. Connect the ONT downlink POTS port to a VoIP phone.

   Make sure the POTS port number matches the one you assigned for voice service with the Smart OMCI web tool. In this example, you can connect either POTS 1 or POTS 2 to the phone.

2. Pick up the phone, you should be able to hear the dial tone and be able to make and receive a phone call.

Delete the OMCI profile

This section describes how to delete the 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, an 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 *2510-tripleplay-me* has one Generic profile, *2510-tripleplay-gen*, and one Specific profile, *1/1/1*, associated with it:

1 Verify ONU 1/1/1 is not active, and the ME profile 2510-tripleplay-me and Generic profile *2510-tripleplay-gen* are associated with ONU 1/1/1.

```
zSH> gpononu show 1/1/1
Slot 1 olt 1
                                        Serial
ONU       Name        Enabled  Model # Number         OMCI files and profiles
=== ================= ======= ======== ====== ================================
  1 1-1-1-1          No                        ME  2510-tripleplay-me
                                             GEN 2510-tripleplay-gen
```

2 Delete the ME profile and Generic profile references to the Specific profile, and delete the Specific profile created on ONU 1/1/1 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 1/1/1 noomci
```

3 Verify the ME profile name and Generic profile name are removed from ONU 1/1/1.

```
zSH> gpononu show 1/1/1
Slot 1 olt 1
                                        Serial
ONU       Name        Enabled   Model # Number         OMCI files and profiles
=== ================ ======= ========= ======= ================================
  1 1-1-1-1          No                        (none)
```

4 Verify the Specific profile on ONU 1/1/1 is also removed.

```
zSH> gpononu profile show spec 1/1/1
```

The outputs does not show 1/1/1 indicating a Specific profile created on 1/1/1 does not exist.

5 Delete the Generic profile, then delete the ME profile.

```
zSH> gpononu profile delete gen 2510-tripleplay-gen

Profile has been deleted!

zSH> gpononu profile delete me 2510-tripleplay-me

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 *2510-tripleplay-me* has one Generic profile, *2510-tripleplay-gen*, and one Specific profile, *1/1/1*, 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 1/1/1

Slot 1 olt 1
                                          Serial
ONU       Name        Enabled   Model #  Number         OMCI files and profiles
=== ================= ======= =========  ============ ========================
  1          1-1-1-1   Yes        2510  ZNTS 1306      ME   2510-tripleplay-me
                                                       GEN  2510-tripleplay-gen

zSH> gpononu status 1/1/1
                         Omci       Gpon    Download    OLT      ONT     Distance
ID    Onu   OperStatus ConfigState OnuStatus  State    Rx Power  Rx Power   (KM)
== ======== ========== =========== ========= ========= ========= ========= =======
1  1-1-1-1     Up     Done         Active    NoUpgrade -23.8 dBm -23.0 dBm   18
```

b Delete the Specific profile created on this ONU.

```
zSH> gpononu profile delete spec 1/1/1
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 1/1/1
                         Omci       Gpon    Download    OLT      ONT     Distance
ID    Onu   OperStatus ConfigState OnuStatus  State    Rx Power  Rx Power   (KM)
== ======== ========== =========== ========= ========= ========= ========= =======
1  1-1-1-1     Up     Config       Active   NoUpgrade -23.8 dBm -23.0 dBm   18
```

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 profile delete me 2510-tripleplay-me
ERROR!  Cannot delete, profile is being used

zSH> gpononu profile delete gen 2510-tripleplay-gen
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 1/1/1 omci
```

Verify the ME profile name and Generic profile name are removed from ONU 1/1/1, and the ONU is disabled.

```
zSH> gpononu show 1/1/1
Slot 1 olt 1
                                          Serial
```

```
ONU       Name        Enabled  Model #  Number         OMCI files and profiles
=== ================= ======= ========  ====== ================================
  1 1-1-1-1          No                      (none)
```

Verify the Specific profile on ONU 1/1/1 is also removed, enter.

```
zSH> gpononu profile show spec 1/1/1
```

The outputs didn't show1/1/1 indicating that the Specific profile does not exist on 1/1/1.

c Delete Generic profile, then delete ME profile.

```
zSH> gpononu profile delete gen 2510-tripleplay-gen

Profile has been deleted!

zSH> gpononu profile delete me 2510-tripleplay-me

Profile has been deleted!
```

Import and export the OMCI profile

Importing the 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, the system will reconcile the associated Generic profile and Specific profile. The 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 profile import me** *meProfileName fileName* command
- **gpononu profile import gen** *genProfileName fileName* command
- **gpononu profile 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 profile show me
me1
me2
```

2 Find the ONUs that use the selected ME profile.

```
zSH> gpononu profile find me me1
This command may take several minutes to complete.
Do you want to continue? <yes or no> [no] yes

1/1/1
```

The above example shows only ONU 1/1/1 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 profile 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 1/1/1

Slot 13  olt 1
                                        Serial
ONU       Name        Enabled   Model #  Number         OMCI files and profiles
=== ================= ======= ========= ============= =======================
  1          1-1-1-1   Yes      2510    ZNTS 1306      ME   2510-tripleplay-me
                                                      GEN  2510-tripleplay-gen
```

The above example shows ONU 1/1/1 uses *2510-tripleplay-gen*. Then update the generic profile *2510-tripleplay-gen* as desired.

```
zSH> gpononu profile update gen 2510-tripleplay-gen
Generic Profile: 2510-tripleplay-gen
   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 [11111]"
  24 "POTS1 Password [11111]"
  25 "POTS2 Dial Number [2222]"
  26 "POTS2 User Name [22222]"
  27 "POTS2 Password [22222]"
  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 profile update spec 1/1/1
Specific Profile: 1/1/1
   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 [11111]"
  24 "POTS1 Password [11111]"
  25 "POTS2 Dial Number [2222]"
  26 "POTS2 User Name [22222]"
  27 "POTS2 Password [22222]"
  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 1/1/1
```

Exporting the 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 profile export me** *meProfileName fileName* command
- **gpononu profile export gen** *genProfileName fileName* command
- **gpononu profile export spec** *slot/olt/onu fileName* command

The following example copies an ME profile file from an MXK-194/198 to another MXK-194/198:

1 On MXK-194/198 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 profile export me me1 2510-mev1.txt
Profile exported.
```

By default, the MXK-194/198 runs as a TFTP server enabling files stored in the *root/pub* folder to be downloaded to other devices with connectivity to the MXK-194/198.

2 On MXK-194/198 B, download the ME profile file *2510-mev1.txt* from the MXK-194/198 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 profile 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 the OMCI profile* on page 357.

Dynamic OMCI GPON zNID installation

This section provides information on how to install OMCI-based GPON zNID with Dynamic OMCI. Figure 37 shows the overall flowchart of the installation procedure. This section includes the following topics:

- Dynamic OMCI overview, page 363
- OMCI GPON zNID installation with Dynamic OMCI, page 376
- View all services on an ONU, page 411
- Delete CPE subscriber profiles on an ONU, page 412
- Move ONU configuration, page 413
- VLAN translation on ONU, page 415

Figure 37: Installation procedure for OMCI GPON zNIDs with Dynamic OMCI

Dynamic OMCI overview

In an effort to improve the MXK-194/198 user experience Zhone has added enhanced functionality to provision CPE devices. The user is now able to use a single macro command to provision a bridge on the MXK-194/198 and the attached CPE at the same time. Voice, Video, and Data services are configured on the MXK-194/198 and flow through to the CPE. This

provisioning method supports on the Zhone GPON ONTs except for ZNID-GPON-42xx and ZNID-GPON-94xx models.

This section describes the following terms used in Dynamic OMCI:

- Internal ME Profiles, page 364
- Dynamic bridging, page 367
- Arbitrary GEM ports, page 371
- CPE Traffic Management Profiles, page 365
- CPE Profiles, page 372

Internal ME Profiles

Figure 38: Internal ME profile in Dynamic OMCI

Zhone provides internal ME profiles for supported Zhone GPON ONTs. The format of a internal ME profile name is zhone-*ZnidModel*.

Internal ME profiles are the indicator of Dynamic OMCI provisioning. As shown in the flowchart step 2c, by specifying an internal ME profile name in the initial setup (use the command **onu set** *OnuInterface* **meprof** *InternalMEProfileName*) on an ONU, the MXK-194/198 knows the model of this ONU, and will provision that ONU with Dynamic OMCI. For example, internal ME profile zhone-5114 defines there are 4 Ethernet ports, 2 POTS ports, and 4 PWE ports on the ZNID-GPON-5114, which supports both SIP and H.248 VoIP signaling.

Zhone also provides a universal ME profile for any zNIDs: zhone-default.

You can use the **gpononu profile show internal-me** command to find the valid internal ME profiles in the MXK-194/198.

```
zSH> gpononu profile show internal-me
zhone-2501
zhone-2504
zhone-2510
zhone-2510a
zhone-2511
zhone-2516
zhone-2517
zhone-2520
```

```
zhone-5114
zhone-5120
zhone-8224
zhone-8324
zhone-cig
zhone-7310
zhone-2424
zhone-default
```

To manage an ONU with Dynamic OMCI, associate an internal ME profile to the ONU with the **gpononu set** *slot/OLT port/ONU port* **meprof** *InternalMEprofile* command.

```
zSH> gpononu set 1/5/1 meprof zhone-5114
```

CPE Traffic Management Profiles

You can use the CPE Traffic management profile to configure the bandwidth shaping on GEM ports. It contains rate control on upstream, priority control on both upstream and downstream direction, and weight control parameters on both direction.

The CPE traffic management profile is optional when creating a GEM port.

Note: Rate control and Priority (weight) control cannot be used on the same ONU. If rate control is applied to one GEM port, it will be enabled for the whole ONU, the other GEM ports in the same ONU must use rate control too. Same for Priority (Weight) control.

Command:

```
cpe traffic add <profile-name>
    [ us-sir < value > ]
    [ us-pir < value > ]
    [ us-priority < value > ]
    [ us-weight < value > ]
    [ ds-priority < value > ]
    [ ds-weight < value > ]
    Create a profile, the <profile-name> must be unique
and the profile index will be automatically generated
```

Table 10: cpe traffic add Command Option Explanations

Command Option	Description
profile-name	Specify a unique name for the CPE traffic management profile. A profile index will be automatically generated after creation of this profile.
us-sir *value*	Upstream sustained information rate, in kilobits per second. Value range is 0 to 1310720.
us-pir *value*	Upstream peak information rate, in kilobits per second. Value range is 0 to 1310720.

Table 10: cpe traffic add Command Option Explanations

Command Option	Description
us-priority *value*	Upstream priority, for the strict priority scheduling policy. Value range is 0 to 7 where where 0 is the highest priority.
us-weight *value*	Upstream weight, for the weighted round robin scheduling policy. Value range is 0 to 255 where 0 is the lowest weight.
ds-priority *value*	Downstream priority, for the strict priority scheduling policy. Value range is 0 to 7 where 0 is the highest priority.
ds-weight *value*	Downstream weight, for the weighted round robin scheduling policy. Value range is 0 to 255 where 0 is the lowest weight.

Configuring traffic shaping on GEM ports with CPE traffic management profiles

The following example shows how to create a CPE traffic management profile for Data subscribers, and associate it to a GEM port.

1. Create a CPE traffic management profile with profile name 2MDataPlan, specify the us-sir and us-pir fields. A profile index will be automatically created after this command.

```
zSH> cpe traffic add 2MDataPlan us-sir 2048 us-pir 3072
Profile “2MDataPlan” has been created with index 1

zSH> cpe traffic show all
                                    upstream upstream upstream downstream upstream downstream
  Index             Profile Name        SIR      PIR    priority  priority  Weight   Weight
========= ================================ ======== ======== ======== ========== ======== ==========
1             2MDataPlan                   2048    3072       0         0        0         0
1 entries found.
```

2. Associate this CPE traffic management profile (tm) to GEM port 501 on ONU 1/1/5. The us-sir and us-pir fields will apply to the GEM port.

   The CPE traffic management profile can be refered to by either profile-name or profile-index.

```
zSH> bridge add 1-1-1-5/gpononu gem 501 gtp 1 tm 2MDataPlan downlink vlan 100
tagged eth 1
Adding bridge on 1-1-1-5/gpononu
Created bridge-interface-record 1-1-1-501-gponport-100/bridge
```

Dynamic bridging

Figure 39: Dynamic bridging in Dynamic OMCII

Dynamic bridging is only applicable to the Dynamic OMCI. It appears as step 4a in the flowchart. It uses a single macro command "**bridge add**" to create both MXK-194/198 bridge and CPE connections, and define the bridge-related parameters for service types.

The MXK-194/198 bridge and CPE connection can have one-to-one and one-to-many mappings.

As shown in Figure 40, the one-to-one mapping is one MXK-194/198 bridge created on a GEM port that maps to one CPE connection created on an ONU UNI port.

Figure 40: The one-to-one mapping between MXK-194/198 bridges and CPE Connections

As shown in Figure 41, the one-to-many mapping is one MXK-194/198 bridge created on a GEM port that maps to multiple CPE connections created on multiple ONU UNI ports.

Figure 41: The one-to-many mapping between MXK-194/198 bridges and CPE Connections

Creating a one-to-many mapping between MXK-194/198 bridge and CPE connections

The following example creates a one-to-many mapping which has one MXK-194/198 bridge to mapping with two CPE connections. These two CPE connections share the same GEM port ID, VLAN, GTP (GPON Traffic Profile) and TM (CPE Traffic Management Profile).

1 Create a MXK-194/198 bridge, and the first CPE connection:

```
zSH> bridge add 1-1-1-5/gpononu gem 501 gtp 1 tm 1 downlink vlan 100 tagged eth 1 uni-vlan 100
Adding bridge on 1-1-1-5/gpononu
Created bridge-interface-record 1-1-1-501-gponport-100/bridge
```

The first part of this command "**`bridge add 1-1-1-5/gpononu gem 501 gtp 1 tm 1 downlink vlan 100 tagged`**" creates a new MXK-194/198 bridge. The second part of this command "**`eth 1 uni-vlan 100`**" creates a CPE connection with this MXK-194/198 bridge. This example is passing single-tagged frames on VLAN 100 as is from Ethernet UNI port 1. For VLAN ID translation on ONU, refer to VLAN translation on ONU, page 415.

GTP is a mandatory field in the **bridge add** command when creating a MXK-194/198 bridge. It contains the bandwidth allocation information for the T-cont. For detail, refer to Bandwidth allocation on T-cont, page 471.

TM is an optional field in the **bridge add** command when creating a MXK-194/198 bridge. It contains the traffic shaping information for the GEM port. for detail, refer to CPE Traffic Management Profiles, page 365.

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN   Physical       Bridge                        St   Table
Data-----------------------------------------------------------------------------------------
dwn            Tagged 100 1/1/1/5/gpononu 1-1-1-501-gponport-100/bridge DWN
upl            Tagged 100 1/1/5/0/eth     ethernet5-100/bridge      UP   S VLAN 100
```

```
default
2 Bridge Interfaces displayed

zSH> bridge show onu
            GEM   ONU     ONU UNI    OLT           OLT
ONU         Port  UNI     VLAN/SLAN  VLAN/SLAN     MVR   Service  OLT Bridge                                ST
-------------------------------------------------------------------------------------------------------------------
1/1/5       501   eth 1        100 Tagged 100       data     1-1-1-501-gponport-100/bridge           DWN
1 Bridge Interfaces displayed
1 GPON ONU Connections displayed
```

2 Create the second CPE connection, and map it to the newly created MXK-194/198 bridge.

```
zSH> bridge add 1-1-1-5/gpononu gem 501 vlan 100 tagged eth 2
GPON ONU Connection 1-1-1-5/gpononu/1/2/0/0 has been created
```

The first part of this command "**`bridge add 1-1-1-5/gpononu gem 501 vlan 100 tagged`**" indicates the MXK-194/198 bridge. The second part of this command "**`eth 2`**" creates another CPE connection with this MXK-194/198 bridge. This example is adding VLAN 100 to untagged data traffics from Ethernet UNI port 2.

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN   Physical       Bridge                        St   Table Data
-------------------------------------------------------------------------------------------
dwn            Tagged 100 1/1/1/5/gpononu 1-1-1-501-gponport-100/bridge DWN
upl            Tagged 100 1/1/5/0/eth     ethernet5-100/bridge          UP   S VLAN 100  default
2 Bridge Interfaces displayed

zSH> bridge show onu
            GEM   ONU     ONU UNI    OLT           OLT
ONU         Port  UNI     VLAN/SLAN  VLAN/SLAN     MVR   Service  OLT Bridge                                ST
-------------------------------------------------------------------------------------------------------------------
1/1/5       501   eth 1        100 Tagged  100         data     1-1-1-501-gponport-100/bridge           DWN
1/1/5       501   eth 2            Tagged  100         data     1-1-1-501-gponport-100/bridge           DWN
1 Bridge Interfaces displayed
    2 GPON ONU Connections displayed
```

Table 11 provides a summary on how to add bridges for supported services.

As shown in the table, for VoIP or PWE services, instead of **keyword + UNI port index**, the **bridge add** command specifies **keyword** only, which indicates the CPE connection is created on all the service-related UNI ports on this ONU.

Table 11: Dynamic bridging for services

Services	Bridges	Service Keywords	Command Examples
Data	One **bridge add** command per CPE connection	N/A Note: when no service keyword is specified, it implies data service.	**`bridge add 1-1-3-1/gpononu gem 301 gtp 1 downlink vlan 100 tagged eth 1`** (create Data service on ethernet port 1 on the ONU)

Table 11: Dynamic bridging for services

Services	Bridges	Service Keywords	Command Examples
Video	One **bridge add** command per CPE connection	video	**`bridge add 1-1-3-1/gpononu gem 401 gtp 1 0/4 downlink vlan 999 tagged video eth 2`** (create Video service on ethernet port 2 on the ONU)
VoIP	One **bridge add** command per ONU	sip, sipplar, or h248	**`bridge add 1-1-3-1/gpononu gem 702 gtp 1 downlink vlan 300 tagged sip`** (create SIP VoIP service on all POTS ports on the ONU) **`bridge add 1-1-3-1/gpononu gem 702 gtp 1 downlink vlan 300 tagged sipplar`** (create SIP PLAR VoIP service on all POTS ports on the ONU) **`bridge add 1-1-3-1/gpononu gem 702 gtp 1 downlink vlan 300 tagged h248`** (create H.248 VoIP service on all POTS ports on the ONU)
PWE	One **bridge add** command per ONU	pwe	**`bridge add 1-1-3-1/gpononu gem 602 gtp 1 downlink vlan 500 tagged pwe`** (create PWE service on all T1/E1 ports on the ONU)

All the command examples shown in Table 11 are adding a VLAN to untagged traffics from ONU UNI ports. For VLAN ID translation, refer to VLAN translation on ONU, page 415.

Deleting MXK-194/198 bridge and associated CPE connections

The following example deletes a one-to-many mapping which has one MXK-194/198 bridge mapping to two CPE connections.

To remove the MXK-194/198 bridge and all the associated CPE connections.

1. To remove the MXK-194/198 bridge and all the associated CPE connections at the same time, you can use this **bridge delete all** command

```
zSH> bridge delete 1-1-1-501-gponport-100/bridge all
   GPON ONU Connection 1-1-1-501-gponport-100/bridge/1/1/100/0 has been deleted
   GPON ONU Connection 1-1-1-501-gponport-100/bridge/1/2/0/0 has been deleted
   1-1-1-501-gponport-100/bridge delete complete
```

2 Or you can remove the CPE connections first, and then remove the MXK-194/198 bridge.

```
zSH> bridge delete 1-1-1-501-gponport-100/bridge eth 1 uni-vlan 100
   GPON ONU Connection 1-1-1-501-gponport-100/bridge/1/1/100/0 has been deleted
```

If there is only one CPE connection associated with the MXK-194/198 bridge, you can delete the MXK-194/198 bridge interface directly.

```
zSH> bridge delete 1-1-1-501-gponport-100/bridge
   GPON ONU Connection 1-1-1-501-gponport-100/bridge/1/2/0/0 has been deleted
   1-1-1-501-gponport-100/bridge delete complete
```

Arbitrary GEM ports

In the Dynamic OMCI, any GEM port ID in the range of 257 to 3828 is allowed to be associated with any ONU. Note that each of these GEM port IDs needs to be unique for the OLT port.

When creating a GEM port with the **bridge add** command, users specify both the *ONU interface ID* and *GEM port ID*:

```
zSH> bridge add 1-1-3-1/gpononu gem 510 gtp 1
downlink vlan 1001 tagged eth 1
Adding bridge on 1-1-3-1/gpononu
Created bridge-interface-record 1-1-3-510-gponport-1001/
bridge
```

If the specified GEM port ID is free, then it will be assigned to the ONU. If the GEM port ID already exists and has been used by the same ONU, it will be reused. If it has been assigned to a different ONU, an error message appears and the command will fail.

To view what GEM port IDs are used in the ONU, use the **gpononu gemports** command.

The **gpononu gemports** command has options to select by slot, OLT, or ONU. If you run the command without defining the slot/OLT/ONU the command will check for ONTs on every port of every card and depending on the number of cards, may take a long time to complete.

```
zSH> onu gemports 1/3/1
                                                Fixed UBR Fixed CBR Assured   Max       Extra
                                   traf         Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU          GEM Port   Admin prof   compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type       allocId DBA
=========== ============ ===== ====== ===== ===== ========= ========= ========= ========= ========== ======= =====
    1-1-3-1  1-1-3-510    Up      1 False False     2.048         0       n/a       n/a        n/a     510 n/a
              1-1-3-710    Up      3 False False         0     0.512       n/a       n/a        n/a     641 n/a
              1-1-3-650    Up      2 False False         0     0.512       n/a       n/a        n/a     640 n/a
```

The **bridge add** command example also defines which VLAN is matched to the GEM port. As shown in Figure 42. Depends on your implementation, users can specify one VLAN for one service, and assign the same VLAN to different ONU GEM ports. In this example, the service provider uses zNID 1 and zNID 42 for businesses, uses zNID 2 and 3 for residential area.

Figure 42: Example GPON VLAN implementation

CPE Profiles

Figure 43: CPE profiles in Dynamic OMCII

CPE profiles define the different services that are provisioned on CPEs. As shown in the flowchart, there are two kinds of CPE profiles:

- CPE shared profiles (used in Step 4b)

  CPE shared profiles contain the common service information which is used by multiple ONU UNI ports.

Note: The CPE shared profile can only be deleted if it is not associated with any other CPE profiles.

- CPE subscriber profiles (used in Step 4c)

  CPE subscriber profiles contain the information for an individual ONU UNI port. When creating a CPE subscriber profile on an ONU UNI port, based on different services, you can associate related CPE shared profiles with it.

  For example, when creating a CPE IP subscriber profile for VoIP service, a CPE IP server profile is required to be associated with it, if there is no CPE IP server profile index specified, the default CPE IP server profile will be used.

All the CPE profiles are listed in Table 12.

As shown in the table, for Data or Video service, the creation of the CPE shared profiles and CPE subscriber profiles are not necessary, unless users want to change the default settings of those profiles. Creation of MXK-194/198 bridges and CPE connections with the **bridge add** command are sufficient for creating a Data or Video service.

Table 12: CPE profile summary

Services	CPE Shared Profiles	CPE Subscriber Profiles
Data	N/A	CPE Eth profile (Optional)
Video	CPE video profile (Optional) CPE video access profile (Optional)	CPE Eth profile (Optional)
VoIP	CPE VoIP server profile CPE VoIP feature profile (Default profile provided. Only for SIP or SIP PLAR) CPE VoIP media profile (Default profile provided) CPE VoIP SIP dialplan profile (Optional. Only for SIP)	CPE VoIP profile (Requires CPE VoIP server profile, CPE VoIP feature profile, and CPE VoIP media profile)
	CPE IP server profile (Default profile provided)	CPE IP profile (Requires CPE IP server profile)
PWE	CPE PWE common profile (Default profile provided)	CPE PWE profile (Requries CPE PWE common profile)
	CPE IP server profile (Default profile provided)	CPE IP profile (Requires CPE IP server profile)

Table 12: CPE profile summary

Services	CPE Shared Profiles	CPE Subscriber Profiles
RF	N/A	CPE RF profile

The step-by-step configuration procedure for each service are provided in *OMCI GPON zNID installation with Dynamic OMCI* on page 376.

Using the CLI Command Menu Shell for CPE profiles

To provision and manage CPE service on ONUs, users can either type a single-line CPE profile macro command, or use the CPE command shell.

The following examples show how to use the CLI command menu shell on CPE profiles.

1. To enter a command shell:

```
zSH> ↵
zSH> cpe↵
zSH> CPE> voip↵
zSH> CPE > VOIP> server↵
zSH> CPE > VOIP> SERVER>
```

2. To get supported commands in the current command shell:

```
zSH> CPE> VOIP> SERVER> ?↵
cpe voip server <add | delete | find | modify | show>
```

3. To get help on the supported commands in the current command shell:

```
zSH> CPE> VOIP> SERVER> help↵

cpe voip server show < index | profile-name | all >
       Display single or multiple profiles

cpe voip server add <profile-name>
           [ primary-server < 64 byte character string > ]
              contains the name ( IP address or resolved name ) of the primary MGC
               SIP proxy server that controls the signalling messages.
           [ secondary-server < 64 byte character string > ]
              contains the name ( IP address or resolved name ) of the secondary or
               backup MGC that controls the signalling messages.
           [ signalling-protocol < sip | siplar | h248 > ]
               specifies the VOIP signalling protocol to use
           [ sip-domain < 64 byte character string > ]
               contains the host or domain part of the SIP address of record for
               users connected to this ONT.
           [ sip-registrar < 64 byte character string > ]
               contains the name ( IP address or resolved name ) of the registrar
               server for SIP signalling messages.
           [ mgc-termination-id-base < 25 byte character string > ]
```

```
                specifies the base string for the H.248 physical termination id's
                for this ONT. This string is intended to uniquely identify an ONT.
                Vendor specific termination identifiers are optionally added to
                this string to uniquely identify a termination on a specific ONT.
            [ oob-dtmf-events < enabled | disabled > ]
               enables or disables handling of DTMF via RTP DTMF events per RFC4733.
            [ softswitch < 4 byte vendor code defined in ANSI T1.220 > ]
               defines the SIP gateway softswitch vendor. The format is four ASCII
               coded alphabetic characters[A..Z] as defined in ANSI T1.220. A value
               of four null characters indicates no particular vendor.
        This command creates a new profile.  The <profile-name> must be supplied
       and must be unique for profile type.  The profile index will be
      automatically generated

cpe voip server modify < index | profile-name > [ arguments ]
       Modify a profile using the profile's <index> or <profile-name>
       See "add" command for available [ arguments ]

cpe voip server delete < index | profile-name >
       Delete a profile using the profile's <index> or <profile-name>

cpe voip server find < index | profile-name >
       Find all cpe-voip-subscriber profiles that reference the cpe-voip-server
```

4 To perform Tab-completions in the current command shell:

```
zSH> CPE> VOIP> SERVER> add metaswitch-sip
primary-server 172.16.60.51 s<Tab>
secondary-server
signalling-protocol
sip-domain
sip-registrar
softswitch

zSH> CPE> VOIP> SERVER> add metaswitch-sip
primary-server 172.16.60.51 se<Tab>

zSH> CPE> VOIP> SERVER> add metaswitch-sip
primary-server 172.16.60.51 secondary-server
```

5 To exit from the current command shell:

```
zSH> CPE> VOIP> SERVER> exit↵
zSH> CPE> VOIP>
```

6 To exit from the CPE command shell:

```
zSH> CPE> VOIP> SERVER> quit↵
zSH>
```

OMCI GPON zNID installation with Dynamic OMCI

In this section, we will provision Data service, Video service, VoIP service, and PWE service on the same ONU, just the MXK-194/198 bridge interface, GEM port setup, GPON traffic profile, VLAN, UNI ports are different. And provision RF service on another ONU. For ease of discussion each of the applications is described separately in this section.

Generally these are the steps to follow to configure the MXK-194/198 to be able to manage OMCI GPON zNIDs with Dynamic OMCI:

- Create High Speed Internet on Dynamic OMCI with Uplink and Downlink, page 376
- Creating uplink and downlink bridges on Dynamic OMCI for video, page 382
- Creating VoIP on Dynamic OMCI on uplink and downlink bridges, page 390
- Creating PWE on Dynamic OMCI on uplink and downlink bridges, page 403
- Creating RF on Dynamic OMCI, page 409

Create High Speed Internet on Dynamic OMCI with Uplink and Downlink

- Creating a GPON traffic profile, page 376
- Specifying internal ME profile for the ONU, page 377

  Only need to specify the internal ME profile once for each ONU.
- Creating uplink/downlink MXK-194/198 bridges, and CPE connections, page 378
- Creating CPE Ethernet subscriber profile (optional), page 379
- Activating the ONT, page 380

  Only need to activate the ONU once.
- Testing the data bridge, page 381

In this example, for data service we will create uplink/downlink bridges with VLAN 1001.

To create data service on the Ethernet UNI ports, use the following steps:

Creating a GPON traffic profile

GPON traffic profiles are a template for defining how traffic will be handled on the bridge with which the GTP is associated. One GTP may be associated with many different bridges. The GTP in this procedure will create a high bandwidth configuration.

Refer to Configure GPON traffic profile, page 472 to get detail configuration and parameter description.

The following is recommended for high speed data configurations.

```
zSH> new gpon-traffic-profile 1
gpon-traffic-profile  1
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 2048
traffic-class: ---------->  {ubr}:     ubr is the default value
compensated: ------------>  {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.
```

Specifying internal ME profile for the ONU

Based on the ONU model, assign the matching internal ME profile to this ONU. In this way it indicates this ONU is provisioned by dynamic OMCI.

1 List the currently valid internal ME profiles.

```
zSH> gpononu profile show internal-me
zhone-2501
zhone-2504
zhone-2510
zhone-2510a
zhone-2511
zhone-2516
zhone-2517
zhone-2520
zhone-5114
zhone-5120
zhone-8224
zhone-8324
zhone-cig
zhone-7310
zhone-2424
zhone-default
```

2 Specify an internal ME profile for an ONU.

After specifying internal ME profile zhone-5114 for ONU 1/3/1, the dynamic OMCI supports are associated with the ONT.

```
zSH> onu set 1/3/1 meprof zhone-5114
```

Creating uplink/downlink MXK-194/198 bridges, and CPE connections

This section will create an uplink and downlink bridges for VLAN 1001 on MXK-194/198, and create CPE connections for the same VLAN on two ONU Ethernet UNI ports:

1. Create an uplink bridge interface on the MXK-194/198

```
zSH> bridge add 1-1-8-0/eth uplink vlan 1001
Adding bridge on 1-1-8-0/eth
Created bridge-interface-record ethernet8-1001/bridge
bridge-path added successfully
```

2. Create downlink bridges on the MXK-194/198 and CPE connections on ONU Ethernet UNI ports.

   This example shows ONU Ethernet UNI port 1 and port 2 share the GEM port 510 and VLAN 1001.

   a. Create an MXK-194/198 bridge on GEM port 510, and a CPE connection on ONU Ethernet UNI port 1. Associate GTP 1 with GEM port 510. Both the CPE connection and the GEM port are in the same VLAN 1001.

```
zSH> bridge add 1-1-3-1/gpononu gem 510 gtp 1 downlink vlan 1001 tagged eth 1
uni-vlan 1001
Adding bridge on 1-1-3-1/gpononu
Created bridge-interface-record 1-1-3-510-gponport-1001/bridge
```

   b. Create the second CPE connection on ONU Ethernet UNI port 2 which are in the same VLAN and share the same GEM port with ONU Ethernet UNI port 1.

```
zSH> bridge add 1-1-3-1/gpononu gem 510 vlan 1001 tagged eth 2 uni-vlan 1001
GPON ONU Connection 1-1-3-1/gpononu/1/2/1001/0 has been created
```

3. View the MXK-194/198 bridges.

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN  Physical        Bridge                          St Table Data
-------------------------------------------------------------------------------
dwn            Tagged 1001 1/1/3/1/gpononu 1-1-3-510-gponport-1001/bridge UP
upl            Tagged 1001 1/1/8/0/eth     ethernet8-1001/bridge          UP S VLAN 1001
default
2 Bridge Interfaces displayed
```

4. View the CPE connections.

```
zSH> bridge show onu
      GEM   ONU   ONU UNI    OLT         OLT
ONU   Port  UNI   VLAN/SLAN  VLAN/SLAN   MVR  Service  OLT Bridge                      ST
-------------------------------------------------------------------------------
1/3/1 510   eth 1 1001/      Tagged 1001      data     1-1-3-510-gponport-1001/bridge   DWN
1/3/1 510   eth 2 1001/      Tagged 1001      data     1-1-3-510-gponport-1001/bridge   DWN
1 Bridge Interfaces displayed
```

```
2 GPON ONU Connections displayed
```

Creating CPE Ethernet subscriber profile (optional)

By default, the admin-state of the CPE Ethernet UNI port is up after creation of CPE connection on that CPE Ethernet UNI port. Because of that, the Data and Video traffic can run on this Ethernet UNI port without further configuration.

If you want to change the default Ethernet physical configurations, you can use the **cpe eth add** command. With this command, the CPE Ethernet subscriber profile is created *manually*. The Ethernet UNI port ID specified in this command must match the one assigned in the **bridge add** command when creating downlink bridge and CPE connection.

Note that a CPE ethernet subscriber profile will also be created *automatically* if users set the ONU UNI ethernet port's admin-state with the **gpononu port** command, or modify the port speed with the **gpononu auto-detect** command. For the details on those command, refer to *Administration of subscriber facing ports* on page 455, and *Configurable speed of subscriber facing ports* on page 456.

After a CPE ethernet subscriber profile is created, if you want to change the settings in that profile, you can use the **cpe eth modify** command, which has the same command syntax as the **cpe eth add** command.

Command:

cpe eth add <*interface*>/<*port number*>

[**admin-state** < **up** | **down** >]

[**rate** < **auto** | **10** | **100** | **1000** >]

[**duplex** < **auto** | **full** | **half** >]

[**mode** < **bridged** | **routed** >]

[**video-profile** < *index* | *profile-name* >]

Create a ETH service. <*interface*> and <*port number*> must be provided.

Table 27 provides the description for command options in the **cpe eth add** command.

Table 13: cpe eth add Command Option Explanations

Command Option	Description
interface/port number	ONU port ID and Ethernet UNI port ID of the physical interfaces.
admin-state *value*	Activates or deactivates the functions performed by the Ethernet port for this subscriber. Possible values are up, down. Default value is up.
rate *value*	Sets the Ethernet port rate. Possible values are auto (default), 10, 100, 1000.
duplex *value*	Sets the Ethernet port duplex. Possible values are auto (default), full, half.

Table 13: cpe eth add Command Option Explanations

Command Option	Description
mode *value*	Specifies whether the Ethernet interface is bridged or derived from an IP router function. Possible values are bridged (default), routed.
video-profile *index* | *profile-name*	Associated CPE video profile. **Note:** The video-profile field is only for video service.

To create a CPE Eth subscriber profile with the **cpe eth add** command:

1. This example changed the Ethernet rate and duplex mode of the Ethernet UNI port 1 on the ONU 1/3/1. Note that this example enters CPE command shell: zSH> CPE> ETH>.

```
zSH> CPE> ETH> add 1/3/1/1 rate 100 duplex full
Service has been created
```

2. Show the settings of the CPE Eth Profile for the Ethernet UNI port 1 on the ONU 1/3/1.

```
zSH> CPE> ETH> show 1/3/1/1
                                                  Video     Traf Mngt
   CPE     Port Number  Admin  Rate Duplex   Mode   Profile   Profile
========== =========== ======= ==== ====== ======== ========= =========
     1/3/1           1      up 100    full bridged         0         0
1 services displayed
```

3. Show all the services created for the ONU.

```
zSH> CPE> show 1/3/1

CPE 1/3/1

Service: DATA
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN Admin Oper
---- ------ ------------- ------------- ----- -----
 510   eth 1     1001/----    Tagged 1001    up  down
 510   eth 2     1001/----    Tagged 1001     up  down
```

Activating the ONT

Activate the ONT to add it to the system. In Dynamic OMCI, after changing the service configuration on an activated ONU, the services configuration will be updated automatically.

1. To activate an ONT first run the **gpononu show** command to display the ONTs currently on the OLT, and discover the available serial numbers.

The **gpononu show** command has options to select by slot and OLT. If you run the command without defining the slot/OLT the command will check for ONTs on every port of every card and depending on the number of cards, may take several minutes to complete.

```
zSH> gpononu show 1/3
Free ONUs for slot 1 olt 3:
    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 1 olt 3:
sernoID   Vendor  Serial Number     sernoID   Vendor  Serial Number
    1      ZNTS    93425008
```

2. Run the **gpononu set** command to associate a serial number to the appropriate ONT:

```
zSH> gpononu set 1/3/1 1
Onu 1 successfully enabled with serial number ZNTS
93425008
```

3. Run the **gpononu show** command to verify the ONT is enabled, and OMCI support is added into the ONT (the associated internal ME profile can be displayed).

```
zSH> gpononu show 1/3/1
                                          Serial
ONU       Name        Enabled Model # Number         OMCI files and profiles
=== ================= ======= ======= ============== ========================
  1           1-1-3-1   Yes       5114 ZNTS 93425008   ME   zhone-5114
Note: NULL Model String indicates not able to get model ID
```

4. Run the **gpononu status** command to verify the OMCI Config State is active.

```
zSH> gpononu status 1/3/1
                          Omci       Gpon    Download    OLT      ONT     Distance
ID    Onu   OperStatus ConfigState OnuStatus  State    Rx Power  Rx Power   (KM)
== ======== ========== =========== ========= ========= ========= ========= =======
1  1-1-3-1     Up     Active      Active    NoUpgrade -19.2 dBm -20.0 dBm   18
```

Testing the data bridge

Verify that the user can get data on the PC:

1. Connect an ONT downlink ethernet port to the PC.

   Make sure the ONT model matches the internal ME profile you assigned. This example connects a ZNID-GPON-5114 to the PC.

And also make sure the ONT downlink ethernet port number matches the one you assigned with the **bridge add** command for data service. In this example, you can connect either ETH 1 or ETH 2 to the PC.

2 Run the **bridge show** command to view the MAC address of the connected PC.

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN Physical        Bridge                        St Table Data
-------------------------------------------------------------------------------
upl           Tagged 1001 1/1/1/5/0/eth  ethernet5-1001/bridge       UP S VLAN 1001 default
dwn           Tagged 1001 1/1/3/1/gpononu 1-1-3-510-gponport-1001/bridge
UP D 00:00:86:43:3c:e4 MAC of PC
```

3 Open a command prompt on the PC and enter **ipconfig** to verify that you can get an IP address from DHCP server for the PC.

4 Open an internet browser on the PC, you should be able to access the internet now.

Creating uplink and downlink bridges on Dynamic OMCI for video

Video bridging is very similar to data bridging, it uses downlink and uplink bridges as well, but the GTP, GEM ports and VLANs are different. During configuring data service, the internal ME profile name is already specified to the ONU, so if the video service is going to be configured on the same ONU, there is no need to specify the internal ME profile again.

To create video service on the Ethernet UNI ports on the same ONU use the following steps:

- Creating GPON traffic profile, page 382
- Creating uplink and downlink bridges, and CPE connections, page 383

  Creation of Video bridge in this step is sufficient for creating a video service. If you want to configure additional common video service attributes and subscriber attributes, you can use the following optional steps.

- Creating CPE video access control (optional), page 384
- Creating CPE video profile and associate it with a CPE video access control list (optional), page 387
- Creating CPE Ethernet subscriber profile and associate it with a CPE video profile (optional), page 389
- Testing the IPTV bridge, page 390

Creating GPON traffic profile

Add the GPON traffic profile.

The following GPON traffic profile is recommended for video:

```
zSH> new gpon-traffic-profile 2
gpon-traffic-profile 2
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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.
```

Creating uplink and downlink bridges, and CPE connections

This section will create an uplink and downlink bridge for VLAN 999:

1 Create an uplink bridge interface

a Create the uplink bridge interface

The following example creates a video uplink bridge interface.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 999 tagged

Adding bridge on 1-1-4-0/eth
Created bridge-interface-record ethernet4-999/
bridge
Bridge-path added successfully
```

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
      1001 ethernet8/bridge                   Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
       999 ethernet4-999/bridge               Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
```

b Modify the bridge-path for the uplink with proxy reporting and 30 seconds igmp query interval. Note how the igmptimer is added to the bridge-path.

```
zSH> bridge-path modify ethernet4-999/bridge vlan
999 default igmpsnooping enable igmptimer 30
```

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
      1001 ethernet8/bridge                   Default, Age: 3600, MCAST Age: 150, IGMP
Query Interval: 70, Flap Mode: Default
```

```
        999 ethernet4-999/bridge                Default, Age: 3600, MCAST Age: 61, IGMP
    Query Interval: 30, IGMP Proxy, Flap Mode: Default
```

2 Create downlink bridge interfaces and creates CPE connections on ONU Ethernet UNI ports.

Create downlink bridges on a GPON port and ONU with VLAN ID and GTP.

You can also specify **video** *m*/*n*. *m* indicates the multicast control list on the *MXK-194/198* bridge, *n* indicates the maximum video streams on the MXK-194/198 bridge. By specifying **video 0/4** in this example you can enable subscriptions up to four video streams on the MXK-194/198 bridge interface without control list checking.

If you want to control multicast control list checking on the *CPE connection*, use the **CPE video access add** command to create CPE video access control profiles.

This example adds VLAN 999 to untagged video traffic.

```
zSH> bridge add 1-1-3-1/gpononu gem 650 gtp 2 downlink cos 6 vlan 999 tagged video
0/4 eth 3
Adding bridge on 1-1-3-1/gpononu
Created bridge-interface-record 1-1-3-650-gponport-999/bridge
```

3 View the MXK-194/198 bridges.

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN  Physical        Bridge                          St Table Data
-----------------------------------------------------------------------------------------------
dwn            Tagged  999  1/1/3/1/gpononu 1-1-3-650-gponport-999/bridge    UP
dwn            Tagged 1001 1/1/3/1/gpononu 1-1-3-510-gponport-1001/bridge UP
upl            Tagged 999  1/a/4/0/eth     ethernet4-999/bridge           UP  S VLAN 999  default
upl            Tagged 1001 1/a/8/0/eth     ethernet5-1001/bridge          UP  S VLAN 1001 default
4 Bridge Interfaces displayed
```

4 View the CPE connections.

```
zSH> bridge show onu
      GEM   ONU   ONU UNI    OLT         OLT
ONU   Port  UNI   VLAN/SLAN  VLAN/SLAN   MVR  Service  OLT Bridge                      ST
-------------------------------------------------------------------------------
1/3/1 510   eth 1 1001/      Tagged 1001      data     1-1-3-510-gponport-1001/bridge   UP
1/3/1 510   eth 2 1001/      Tagged 1001      data     1-1-3-510-gponport-1001/bridge   UP
1/3/1 650   eth 3            Tagged 999       iptv     1-1-3-650-gponport-999/bridge    UP
2 Bridge Interfaces displayed
3 GPON ONU Connections displayed
```

Creating CPE video access control (optional)

CPE video access control profile creates an access control list, which defines which multicast addresses the remote-end video can access. That access control list can be specified later in the CPE video profile. If there is no CPE video access control profiles specified in the CPE video profile, there will be no control on the multicast addresses.

Note: The CPE video access control profile can not be deleted if this profile is the only entry in an access control list that is being associated with a CPE video profile.

To create a new CPE video access profile, use this command:

Command:

cpe video access add <*list-name*> [**type <normal | always-on | periodic>**

[**src-ip** <*IPAddress*>]

[**dest-ip-start** <*IPAddress*>]

[**dst-ip-end** <*IPAddress*>]

[**imputed-group-bw** <*value in bytes/sec*>]

Table 27 provides the description for command options in the **cpe video access add** command.

Table 14: cpe video access add Command Option Explanations

Command Option	Description
list-name	CPE video access control profile name.
type	Defines the video stream type. Possible values are normal (default), always-on, periodic.
src-ip *IPAddress*	Source IP address. The default value is 0.0.0.0, that indicates that source IP address is to be ignored.
dst-ip-start *IPAddress*	Destination IP address of the start of the multicast range.
dst-ip-end *IPAddress*	Destination IP address of the end of the multicast range.
imputed-group-bw *value*	Imputed group bandwidth. In the unit of bytes/second. The imputed group bandwidth is used to decide whether or not to honor a join request in the presence of a max multicast bandwidth limit. The default value 0 effectively allows this table entry to avoid maximum bandwidth limitations.

This example creates two CPE video access control profiles, each profile is an entry of a CPE video access control list:

1. Create a CPE video access control profile.

```
zSH> CPE> VIDEO> ACCESS> add basic-plan dst-ip-start
224.10.10.1 dst-ip-end 224.10.10.15 imputed-group-bw
4000
Profile has been created with index 1/1
```

   The first CPE video access control profile in the system is created automatically with list-index 1/entry-index 1.

2. Create the second CPE video access control profile under the same list (1/2):

```
zSH> CPE> VIDEO> ACCESS> add basic-plan dst-ip-start
224.11.10.1 dst-ip-end 224.11.10.4 imputed-group-bw
4000
Profile has been created with index 1/2
```

3 View the two cpe-video-access-control profiles that under the same list. You can either specify list-name or list-index.

```
zSH> CPE> VIDEO> ACCESS> show 1
List/Entry Index          Profile Name                 dstIpStart      dstIpEnd     imputedGroupBw
================ ================================== =============== =============== ===============
           1/1                            basic-plan 224.10.10.1    224.10.10.15            4000
           1/2                            basic-plan 224.11.10.1     224.11.10.4            4000
2 entries found.
```

4 If users want to delete a cpe-video-access-control profile, use the **cpe video access delete** <*list-name* | *list-index*> / <*entry-index*> command.

```
zSH> CPE> VIDEO> ACCESS> delete 1/1
Profile has been deleted.
```

5 If users want to delete the last CPE video access control profile in an access control list that is being associated with a CPE video profile. Users have to remove the reference in the CPE video profile first, and then delete the CPE video access control profile.

This example assumes access control list 1 is being associated with CPE video profile 1, and CPE video access control profile 1/2 is the only entry in the access control list 1.

a Cannot delete CPE video access control profile 1/2.

```
zSH> CPE> VIDEO> ACCESS> delete 1/2
Error:  Cannot delete cpe-video-access-control profile
Cannot delete last entry in a list that is being used by a cpe-video profile.
```

b Find associated CPE video profiles with access control list 1.

```
zSH> CPE> VIDEO> ACCESS> find 1
cpe-video 1/4095/0
1 profiles displayed.
```

c Show the detail information of the associated CPE video profile.

```
zSH> CPE> VIDEO> show 1
                                          Max-Simultaneous Max Mcast Bw      Access
  Index              Profile Name         Groups           Bw        Enforce Control List
========== ================================ ================ ========= ======= ============
         1                       basic-plan                4     50000    true            1
1 entries found.
```

d Remove the reference of the access control list from the CPE video profile. The command option of the **cpe video modify** command is same as **cpe video add**, refer to the next section for the detail.

```
zSH> CPE> VIDEO> modify 1 access-control-list 0
Profile has been modified.
```

e Now we can delete CPE video access control profile 1/2.

```
zSH> CPE> VIDEO> ACCESS> delete 1/2
Profile has been deleted.
```

Creating CPE video profile and associate it with a CPE video access control list (optional)

This step is optional. Creation of Video bridge in step *Creating uplink and downlink bridges, and CPE connections* on page 383 is sufficient for creating a video service. You can use the cpe-video-access-control profile and cpe-video profile, if you want to add some common video service attributes.

Note: CPE video profile can only be deleted when it is not associated with any CPE ethernet subscriber profiles.

Command:

```
cpe video add <profile-name <string>>
          [ max-simultaneous-groups < value > ]
          [ max-mcast-bw < value > ]
          [ bw-enforce < value > ]
          [ igmp-version < value > ]
          [ igmp-function < value > ]
          [ immediate-leave < value > ]
          [ upstream-igmp-rate < value > ]
          [ robustness < value > ]
          [ access-control-list < value > ]
```

Table 27 provides the description for command options in the **cpe video add** command.

Table 15: cpe video add Command Option Explanations

Command Option	Description
profile-name <string>	Specifies a unique CPE video profile name. 36 characters string.
max-simultaneous-groups *value*	Specifies the maximum number of dynamic multicast groups that may be joined by at any one time. Default: **0. Specifies that no administrative limit is to be imposed.**
max-mcast-bw *value*	Specifies the maximum imputed dynamic bandwidth, in bytes per second, that may be delivered to the client port at any one time. Default: **0 . Specifies that no administrative limit is to be imposed**

Table 15: cpe video add Command Option Explanations

Command Option	Description
bw-enforce *value*	Values: **true** Specifies that such attempts be counted and denied. The imputed bandwidth value is taken from the dynamic access control list table, both for a new join request and for pre-existing groups. **false** Specifies that attempts to exceed the max multicast bandwidth be counted but honored. Default: **false**
igmp-version *value*	Specifies the version of IGMP to be supported. Values: **version1** **version2** **version3** Default: **version2**
igmp-function *value*	Enables an IGMP function - transparent IGMP snooping only, snooping with proxy reporting, or IGMP proxy. The function must be consistent with the capabilities specified by the other IGMP configuration attributes. Values: **transparent-snooping** **snoop-with-proxy** **proxy**
immediate-leave *value*	Enables or disables the immediate leave function. Values: **enabled** **disabled** Default: **disabled**
upstream-igmp-rate *value*	Limits the maximum rate of upstream IGMP traffic. Traffic in excess of this limit is silently discarded. The attribute value is specified in messages/second. Default: **0. Imposes no rate limit on this traffic**
robustness *value*	It allows tuning for possible packet loss in the network. Default: **0. It causes the ONT to follow the IETF recommendation to copy the robustness value from query messages originating further upstream.**
access-control-list *value*	This attribute points to a access control group list. Default: **0. It indicates no control list is used.**

1 Create the cpe-video profile.

And you can also associate a video access control list to it.

```
zSH> CPE> VIDEO> add basic-plan max-simultaneous-groups 4 max-mcast-bw 50000
bw-enforce true access-control-list basic-plan
Profile "basic-plan" has been created with index 1
```

2 Show all the cpe-video profiles.

```
zSH> CPE> VIDEO> show all
                                          Max-Simultaneous Max Mcast Bw      Access
   Index              Profile Name         Groups           Bw        Enforce Control List
========== ================================ ================ ========= ======= ============
         1                       basic-plan                4     50000    true            1
1 entries found.
```

3 If users want to delete a cpe video profile, use the **cpe video delete** *<profile-index>* | *<profile-name>* command.

```
zSH> CPE> VIDEO> delete 1
Profile has been deleted.
```

4 Users can use the find command to find the associated CPE Ethernet subscriber profile.

This example assumes CPE video profile 1 is being associated with a CPE ethernet subscriber profile on ONU 1/3/1:

```
zSH> CPE> VIDEO> find 1
cpe-eth-subscriber 1-1-3-1/gpononu
1 profiles displayed
```

Creating CPE Ethernet subscriber profile and associate it with a CPE video profile (optional)

If you want to configure subscriber video service attributes on the Ethernet UNI port, you can use the cpe-eth-subscriber profile. In the cpe-eth-subscriber profile you can change the default Ethernet physical configurations, such as loopback, rate, duplex, or assign a video profile to the Ethernet UNI port. If video profile field is not specified or specified to index 1, the default CPE video profile will be used

For general information and value description of the CPE Ethernet subscriber profile, refer to *Creating CPE Ethernet subscriber profile (optional)* on page 379.

To create a CPE ether subscriber profile for video service:

1 Create a CPE Eth subscriber profile on ONU 1/3/1 Ethernet UNI port 3, and associate it with cpe-video profile “basic-plan”.

Make sure the Ethernet UNI port matches the port ID assigned during the creation of the downlink bridge and CPE connection.

```
zSH> CPE> ETH> add 1/3/1/3 video-profile basic-plan
Service has been created
```

2 Show CPE Eth subscriber profile 1/3/1/3.

```
zSH> CPE> ETH> show 1/3/1/3
                                                Video     Traf Mngt
   CPE      Port Number  Admin  Rate Duplex   Mode    Profile   Profile
========== =========== ======= ==== ====== ======== ========= =========
```

```
     1/3/1           3      up auto   auto  bridged         1        0
1 services displayed
```

Testing the IPTV bridge

Since we are using a PC and software to emulate a set top box (STB), we can ping out to the video server.

1. Connect an ONT downlink ethernet port to the customer video equipment. This example connects to a PC that runs a STB emulation software.

   Make sure the ethernet port number matches the one you assigned with the **bridge add** command for video service. In this example you can connect ETH UNI port 3 to the PC.

2. Open a command prompt on the PC and enter **ipconfig** to verify that you can get an IP address for the PC.

3. Ping the video server

   a. Open a DOS window

   b. Ping the upstream gateway (provided in your environment setup)

4. Open the STB emulation software and connect to the video server

   As long as you can ping you are showing that you have a data path through the zNID and the MXK-194/198 to the video server. You should be able to connect to the video stream with the STB emulation software.

Creating VoIP on Dynamic OMCI on uplink and downlink bridges

For VoIP service we recommend using uplink and downlink bridge configuration.

There are different types of VoIP signaling: SIP, SIP PLAR, and H.248. One ONU can only use one VoIP signaling protocol.

Note: One ONU only can have one VoIP signaling. For example, if you configured POTS 1 for SIP, all the POTS ports in the same ONU must use SIP too.

Note: Make sure country code in the system profile is set properly for the voice signaling type.

To create voice service on the POTS ports on the same ONU, use the following steps:

- Creating GPON traffic profile, page 391
- Creating the uplink and downlink bridge and CPE connection, page 391
- Creating a CPE IP server profile, page 392

- Creating a CPE IP profile for the VoIP service and associate it with a CPE IP server profile, page 393
- Creating CPE VoIP server profiles, page 394
- Creating CPE SIP dial plans for a SIP VoIP server(optional), page 396

  This profile is only needed for SIP voice signaling.
- Creating CPE VoIP features profile for SIP or SIP PLAR, page 397

  This profile is only needed for SIP or SIP PLAR voice signaling.
- Creating CPE VoIP media profile, page 399
- Creating a CPE VoIP subscriber profile and associate it with a VoIP server, a VoIP features profile, and a media profile, page 401
- Testing the VoIP configuration, page 403

Creating GPON traffic profile

Add the GPON traffic profile.

The following GPON traffic profile is recommended for up to four VoIP phones or four POTS ports.

```
zSH> new gpon-traffic-profile 3

gpon-traffic-profile  3
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 512
traffic-class: ---------->  {ubr}:
compensated: ------------>  {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..
```

Creating the uplink and downlink bridge and CPE connection

This section will create a uplink bridge and a downlink bridge on the MXK-194/198 and create CPE connection on the ONU for SIP service:

1 Create a uplink bridge on the uplink interface.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 300 tagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record ethernet4-300/bridge
Bridge-path added successfully
```

2 Create a TLS bridge on the downlink interface and create a CPE connection for all POTS ports on the ONU.

This example creates a TLS MXK-194/198 bridge on GEM port 710, and creates SIP service for all the POTS ports on ONU 1/3/1 and inserts single tagged VLAN 300 to the untagged SIP packets.

```
zSH> bridge add 1-1-3-1/gpononu gem 710 gtp 3 tls vlan 300 tagged sip
Adding bridge on 1-1-3-1/gpononu
Created bridge-interface-record 1-1-3-710-gponport-300/bridge
```

3 On MXK-194/198, run the **bridge show** command to view the MXK-194/198 bridges.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-----------------------------------------------------------------------------------------------------------
dwn             Tagged  300    1/1/3/1/gpononu           1-1-3-710-gponport-300/bridge  UP
dwn             Tagged  999    1/1/3/1/gpononu           1-1-3-650-gponport-999/bridge  UP
dwn             Tagged  1001   1/1/3/1/gpononu           1-1-3-510-gponport-1001/bridge UP
upl             Tagged  300    1/1/4/0/eth               ethernet4-300/bridge           UP
upl             Tagged  999    1/1/4/0/eth               ethernet4-999/bridge           UP  S VLAN 999 default
upl             Tagged  1001   1/1/8/0/eth               ethernet8/bridge               UP  S VLAN 1001 default
6 Bridge Interfaces displayed
```

4 View the CPE connections.

```
zSH> bridge show onu
           GEM   ONU     ONU UNI    OLT           OLT
ONU        Port  UNI     VLAN/SLAN  VLAN/SLAN     MVR   Service  OLT Bridge                                ST
-----------------------------------------------------------------------------------------------------------
1/3/1      510   eth 1   1001/----  Tagged  1001        data     1-1-3-510-gponport-1001/bridge          UP
1/3/1      510   eth 2   1001/----  Tagged  1001        data     1-1-3-510-gponport-1001/bridge          UP
1/3/1      650   eth 3              Tagged  999         iptv     1-1-3-650-gponport-999/bridge           UP
1/3/1      710   pots               Tagged  300         sip      1-1-3-710-gponport-300/bridge           UP
3 Bridge Interfaces displayed
4 GPON ONU Connections displayed
```

Creating a CPE IP server profile

The default CPE IP server profile specified the DHCP as the host IP option. It indicates CPE will get the host IP address automatically from the DHCP server.

Note: CPE IP server profile can only be deleted when it is not associated by any CPE IP profiles.

Command:

cpe ip server add <*profile-name*>

[**host-ip-option** < **dhcp** | **static** >]

[**netmask** < *value* >]

[**gateway** < *IP address* >]

[**primary-dns** < *IP address* >]

[**secondary-dns** < *IP address* >]

Table 16: cpe ip server add Command Option Explanations

Command Option	Description
profilename	Specifies a unique CPE IP server profile.
host-ip-option <**dhcp**| **static** >	Selects an IP related option. DHCP is the default value. It indicates CPE will get the host IP address automatically from the DHCP server.
netmask <*value*>	Specifies the subnet mask for IP host services.
gateway <*IP address*>	Specifies the default gateway address used for IP host services, this attribute has default value 0.0.0.0.
primary-dns <*IP address*>	Specifies the primary DNS IP address. If this value is 0.0.0.0, no primary SIP DNS is defined. The default value is 0.0.0.0.
secondary-dns <*IP address*>	Specifies the secondary DNS IP address. If this value is 0.0.0.0, no second SIP DNS is defined. The default value is 0.0.0.0.

The following example creates a static CPE IP server profile:

1 Create a CPE IP server profile with profile-name. The profile index will be generated automatically.

```
zSH> CPE> IP> SERVER> add IPserver host-ip-option static netmask 255.255.255.0
gateway 172.168.3.254 primary-dns 172.168.19.1
Profile "IPserver" has been created with index 2
```

2 Show the default CPE IP server and user-created CPE IP servers.

```
zSH> CPE> IP> SERVER> show all
                                                  host IP
   Index              Profile Name                 option    netmask          gateway       primary DNS    secondary DNS
========== ================================== ====== =============== =============== =============== ==============
1          Default_Cpe_Ip_Server                    dhcp   255.255.255.0         0.0.0.0         0.0.0.0         0.0.0.0
2          IPserver                               static   255.255.255.0   172.168.3.254    172.168.19.1         0.0.0.0
2 entries found.
```

3 If you want to delete a user-created CPE IP server, use the delete command with the profile index or profile name.

```
zSH> CPE> IP> SERVER> delete IPserver
Profile has been deleted.
```

Creating a CPE IP profile for the VoIP service and associate it with a CPE IP server profile

Create a CPE IP profile for the VoIP service and associate it with a CPE IP server. If there is no CPE IP server specified, the default CPE IP server will be used.

Command:

cpe ip add <*interface*> <**voip**|**pwe**>

[**host-ip** < *IP address* >]

[**ip-server** < *index* | *profile-name* >]

Table 17: cpe ip add Command Option Explanations

Command Option	Description
interface	Specifies an ONU interface ID.
<**voip** | **pwe** | **wifi** >	Specifies a service type.
host-ip <*IP address*>	Specifies the address used for IP host services. The default value is 0.0.0.0.
ip-server <*index* | *profile-name*>	Associates a CPE IP server profile with this host IP. If this field is not specified or is 1, the default CPE IP server profile (index 1) with DHCP enabled will be used.

1. Create a CPE IP profile for VoIP service.

```
zSH> CPE> IP> add 1/3/1 voip
Service has been created
```

The default CPE IP server with index 1 will be used.

2. Show CPE IP profiles.

```
zSH> CPE> IP> show all
   CPE     Service     host IP     IP Server
========== ======= =============== =========
      1/3/1 VOIP         0.0.0.0         1
1 services displayed
```

3. Delete a CPE IP profile.

```
zSH> CPE> IP> delete 1/3/1 voip
Service has been deleted.
```

Creating CPE VoIP server profiles

This section will create a CPE VoIP server profile. Each CPE VoIP server profile specifies the IP addresses or names of the primary and secondary VoIP servers, and the VoIP signalling protocol. The VoIP signalling protocol could be SIP, SIP PLAR, or H.248. This example sets it to SIP.

Note: CPE VoIP server profile can only be deleted when it is not associated by any CPE VoIP subscriber profiles.

Command:

cpe voip server add <*profile-name*>

[**primary-server** < *64 byte character string* >]

[**secondary-server** < *64 byte character string* >]

[**signalling-protocol** < **sip** | **siplar** | **h248** >]

[**sip-domain** < *64 byte character string* >]

[**sip-registrar** < *64 byte character string* >]

[**mgc-termination-id-base** < *25 byte character string* >]

[**oob-dtmf-events** < **enabled | disabled** >]

[**softswitch** < *4 byte vendor code defined in ANSI T1.220* >]

Table 27 provides the description for command options in the **cpe voip server add** command.

Table 18: cpe voip server add Command Option Explanations

Command Option	Description
profilename	Specifies a unique CPE VoIP server profile.
primary-server *value*	contains the name (IP address or resolved name) of the primary MGCP or SIP proxy server that controls the signalling messages.
secondary-server *value*	contains the name (IP address or resolved name) of the secondary or backup MGCP proxy server that controls the signalling messages.
signalling-protocol < **sip** | **siplar** | **h248** >	specifies the VoIP signalling protocol
sip-domain < 64 byte character string >	contains the host or domain part of the SIP address of record for users connected to this ONT.
sip-registrar < 64 byte character string >	contains the name (IP address or resolved name) of the registrar server for SIP signalling messages.
mgc-termination-id-base < 25 byte character string >	specifies the base string for the H.248 physical termination id's for this ONT. This string is intended to uniquely identify an ONT. Vendor specific termination identifiers are optionally added to this string to uniquely identify a termination on a specific ONT.
oob-dtmf-events < **enabled** | **disabled** >	enables or disables handling of DTMF via RTP DTMF events per RFC 4733.
softswitch < 4 byte vendor code defined in ANSI T1.220 >	defines the SIP gateway softswitch vendor. The format is four ASCII coded alphabetic characters[A..Z] as defined in ANSI T1.220. A value of four null characters indicates no particular vendor.

To create VoIP server profiles, use the **cpe voip server add** command.

1 This example creates a VoIP server profile for a SIP VoIP server.

```
zSH> CPE> VOIP> SERVER> add metaswitch-sip primary-server 172.16.60.51
signalling-protocol sip sip-domain metaswitch.oak.zhone.com sip-registrar
metaswitch.oak.zhone.com
Profile "metaswitch-sip" has been created with index 1
```

2 Shows the VoIP server profile.

```
zSH> CPE> VOIP> SERVER> show metaswitch-sip
                                        Signalling     Mgc Termination       Oob Dtmf
 Index         Profile Name              Protocol         Id Base             Events
======= ============================== ========== ========================= ========
      1                 metaswitch-sip        sip                           disabled
```

```
 Primary Server   : 172.16.60.51
 Secondary Server :
 Sip Domain       : metaswitch.oak.zhone.com
 Sip Registrar    : metaswitch.oak.zhone.com
 Softswitch       :
1 entries found.
```

Creating CPE SIP dial plans for a SIP VoIP server(optional)

CPE SIP dialplans are only for SIP. You can create up to 30 CPE SIP dialplans for each CPE SIP VoIP server.

Command:

cpe voip dialplan add < *server-index* | *server-profile-name* >

[**dial-plan-format** < **h248** | **nsc** | **vendor-specific** >]

[**dial-plan** < *25 character string* >]

Table 19: cpe sip dialplan add Command Option Explanations

Command Option	Description
profilename	Specifies a unique CPE VoIP server profile name.
dial-plan-format <h248| nsc | vendor-specific >	Defines the dialplan format standard that is supported on the ONT for VoIP service. It could be h248, nsc, or vendor-specific. The default value is h248.
dial-plan < *25 character string* >	Defines the dialplan used by the VoIP service.

1. Create the first CPE SIP dialplan profile for SIP VoIP server 1.

```
zSH> CPE> VOIP> DIALPLAN> add 1 dial-plan 1xx|[2-7]xxxxxx
Profile has been created with index 1/1
```

2. Create the second CPE SIP dialplan profile for the same SIP VoIP server 1.

   The vertical bar in this example are entered by pressing Shift and backsplash keys together.

```
zSH> CPE> VOIP> DIALPLAN> add 1 dial-plan xx.T|*xx.T
Profile has been created with index 1/2
```

3. Show all the SIP dialplans.

```
zSH> CPE> VOIP> DIALPLAN> show all
   index      dial-plan-format        dial-plan
============= ================ =========================
         1/1             h248           1xx|[2-7]xxxxxx
         1/2             h248                xx.T|*xx.T
2 entries found.
```

Creating CPE VoIP features profile for SIP or SIP PLAR

Create a CPE VoIP features profile for the VoIP service. The VoIP features profiles are using the bitmaps format.

Note: The CPE VoIP features profile is only applicable for SIP or SIP PLAR VoIP server.

Note: CPE VoIP feature profile can only be deleted when it is not associated by any CPE VoIP subscriber profiles.

Command:

cpe voip features add <*profile-name*>

[announcement-type < silence | reordertone | fastbusy | voice | na >]

[cid-features < calling-number | calling-name | cid-block | cid-number | cid-name | anonym-block | all | none >]

[call-waiting-features < call-waiting | cid-announcement | all | none >]

[call-progress-or-transfer-features < 3-way | call-transfer | call-hold | call-park | do-not-disturb | flash-on-emergency | emergency-hold | 6-way | all | none >]

[call-presentation-features < msg-wait-splash-ring | msg-wait-special-dial-tone |msg-wait-visual | call-fwd | all | none >]

Table 20: cpe voip features add Command Option Explanations

Command Option	Description
profilename	Specifies a unique CPE voip features profile.
announcement-type < silence | recordertone| fastbusy | voice| na	specifies the treatment when a subscriber goes off hook but does not attempt a call.
cid-features <calling-number| calling-name| cid-blocking| cid-number | cid-name |anonym-block | all | none>	Specifies the bit map of the caller ID features.
call-waiting-features < call-waiting | cid-announcement | all | none >	Specifies the bit map of the call waiting features.
call-progress-or-transfer-features < 3-way | call-transfer | call-hold | call-park | do-not-disturb | flash-on-emergency | emergency-hold | 6-way | all | none >	Specifies the bit map of the call processing features.

Table 20: cpe voip features add Command Option Explanations

Command Option	Description
call-presentation-features < msg-wait-splash-ring | msg-wait-special-dial-tone | msg-wait-visual | call-fwd | all | none >	Specifies the bit map of call presentation features.

1. Add a CPE VoIP features profile. By only specifying profile-name field, the profile will be created with all the default settings.

```
zSH> CPE> VOIP> FEATURES> add featureslist1
Profile “featurelist1” has been created with index 2
```

2. Show all the CPE VoIP features profiles, including the default and the user-created profiles.

```
zSH> CPE> VOIP> FEATURES> show all

cpe-voip-features 1       Name: Default_Cpe_Voip_Features       Type: reordertone
  Caller ID        Call Waiting     Call Progress or        Call Presentation
   Features          Features       Transfer Features            Features
==============   ================   ==================   =========================
calling-number   call-waiting       3-way                msg-wait-splash-ring
calling-name     cid-announcement   call-transfer       msg-wait-special-dial-tone
cid-blocking                        call-hold            msg-wait-visual
cid-number                          call-park            call-fwd
cid-name                            do-not-disturb
anonym-block                        flash-on-emergency
                                    emergency-hold
                                    6-way

cpe-voip-features 2       Name: featurelist1                    Type: reordertone
  Caller ID        Call Waiting     Call Progress or        Call Presentation
   Features          Features       Transfer Features            Features
==============   ================   ==================   =========================
calling-number   call-waiting       3-way                msg-wait-splash-ring
calling-name     cid-announcement   call-transfer       msg-wait-special-dial-tone
cid-blocking                        call-hold            msg-wait-visual
cid-number                          call-park            call-fwd
cid-name                            do-not-disturb
anonym-block                        flash-on-emergency
                                    emergency-hold
                                    6-way
```

3. Modify the CPE VoIP features profile.
   - To clear a bit map value, place a minus sign infront of the argument. For example: “-calling-name” clears the calling-name value in the cid-features.
   - To clear all features in a bit-map, use the “none” keyword.
   - To enable all features in a bit-map, use the “all” keyword.

```
zSH> CPE> VOIP> FEATURES> modify featurelist1 cid-features -calling-name
call-waiting-features none call-progress-or-transfer-features all
call-presentation-features -call-fwd
Profile has been modified.
```

4 Show the CPE VoIP features profile.

```
zSH> CPE> VOIP> FEATURES> show featurelist1
cpe-voip-features 2       Name: featurelist1                    Type: reordertone
  Caller ID        Call Waiting     Call Progress or        Call Presentation
   Features          Features       Transfer Features            Features
==============   ================   ==================   =========================
calling-number                      3-way                msg-wait-splash-ring
cid-blocking                       call-transfer        msg-wait-special-dial-tone
cid-number                          call-hold            msg-wait-visual
cid-name                            call-park
anonym-block                        do-not-disturb
                                    flash-on-emergency
                                    emergency-hold
                                    6-way
1 entries found.
```

Creating CPE VoIP media profile

Create a CPE VoIP media profile for the VoIP service.

Note: CPE VoIP media profile can only be deleted when it is not associated by any CPE VoIP subscriber profiles.

Command:

cpe voip media add <*profile-name*>

[**echo-cancel** < **enabled** | **disabled** >]

[**fax-mode** < **pass-through** | **t38** >]

[**codec-selection-1** < **pcmu** | **gsm** | **g723** | **dvi4-8** | **dvi4-16** |**lpc** | **pcma** | **g722** | **l16.2** |**l16.1** | **qcelp** | **cn** | **mpa** |**gy28** | **dvi4-22** | **g729** >]

[**packet-period-selection-1** < 10 .. 30 >]

[**silence-suppression-1** < **enabled** | **disabled** >]

[**codec-selection-2** < **pcmu** | **gsm** | **g723** | **dvi4-8** | **dvi4-16** | **lpc** | **pcma** | **g722** | **l16.2** |**l16.1** | **qcelp** | **cn** | **mpa** | **gy28** | **dvi4-22** | **g729** >]

[**packet-period-selection-2** < 10 .. 30 >]

[**silence-suppression-2** < **enabled** | **disabled** >]

```
[ codec-selection-3 < pcmu | gsm | g723 |
dvi4-8 | dvi4-16 | lpc | pcma | g722 | l16.2 |l16.1
| qcelp | cn | mpa | gy28 | dvi4-22 | g729 > ]
[ packet-period-selection-3 < 10 .. 30 > ]
[ silence-suppression-3 < enabled |
disabled > ]
[ codec-selection-4 < pcmu | gsm | g723 |
dvi4-8 | dvi4-16 | lpc | pcma | g722 | l16.2 |l16.1
| qcelp | cn | mpa | gy28 | dvi4-22 | g729 > ]
[ packet-period-selection-4 < 10 .. 30 > ]
[ silence-suppression-4 < enabled |
disabled > ]
```

Table 21: cpe voip media add Command Option Explanations

Command Option	Description
profilename	Specifies a unique CPE voip media profile.
echo-cancel < enabled | disabled >	Turns on or off echo cancellation
fax-mode < pass-through | t38 >	Selects the fax mode
codec-selection-n < pcmu | gsm | g723 | dvi4-8 | dvi4-16 |lpc | pcma | g722 | l16.2 |l16.1 | qcelp | cn | mpa |gy28 | dvi4-22 | g729 >	Specifies the codec selection as defined by RFC 3551. n is in the range of 1 to 4.
packet-period-selection-n < 10 .. 30 >	Packet period selection interval in milliseconds. n is in the range of 1 to 4.
silence-suppression-n < enabled | disabled >	Specifies whether silence suppress is on or off. n is in the range of 1 to 4.

1. Create a CPE VoIP media profile.

```
zSH> CPE> VOIP> MEDIA> add T38fax fax-mode t38
Profile "T38fax" has been created with index 2
```

2. Show CPE VoIP media profiles.

```
zSH> CPE> VOIP> MEDIA> show all
                                                  echo                      codec          Packet-period   silence
   Index              Profile Name                cancel    fax mode        Selection      selection       suppression
========== ================================ ======== ========== ================== ============== ==============
1          Default_Cpe_Voip_Media             enabled  passThrough          PCMU (1st)       10 (1st) disabled (1st)
                                                                             PCMU (2nd)       10 (2nd) disabled (2nd)
                                                                             PCMU (3rd)       10 (3rd) disabled (3rd)
                                                                             PCMU (4th)       10 (4th) disabled (4th)
2          T38fax                             enabled  T38                 PCMU (1st)       10 (1st) disabled (1st)
                                                                             PCMU (2nd)       10 (2nd) disabled (2nd)
                                                                             PCMU (3rd)       10 (3rd) disabled (3rd)
                                                                             PCMU (4th)       10 (4th) disabled (4th)

2 entries found.
```

Creating a CPE VoIP subscriber profile and associate it with a VoIP server, a VoIP features profile, and a media profile

To create a CPE VoIP subscriber profile on ONU POTS ports, use the **cpe voip add** command. When creating a CPE VoIP subscriber profile, user must specify a VoIP server profile, associate the VoIP server information to the POTS port. There is no default VoIP server profile. A VoIP features profile and a VoIP media profile are also required when creating the CPE VoIP subscriber profile, if users do not specify these two profiles, then the default profiles are used.

Note that a CPE VoIP subscriber profile will also be created *automatically* if users set the ONU UNI POTS port's admin-state with the **gpononu port** command. For the details, refer to *Administration of subscriber facing ports* on page 455.

After a CPE VoIP subscriber profile is created, to change the settings in that profile, you can use the **cpe voip modify** command, which has the same command syntax as the **cpe voip add** command.

Command:

`cpe voip add` <interface>[/<port number>]

[**`admin-state`** < up | down >]

[**`dial-number`** < 36 byte character string >]

[**`username`** < 25 byte unique character string >]

[**`password`** < 25 byte character string >]

[**`rx-gain`** < -12db - 6db >]

[**`tx-gain`** < -12db - 6db >]

[**`voip-server-profile`** <index | profile-name>]

[**`voip-features-profil`**e <index | profile-name>]

[**`voip-media-profile`** <index | profile-name>]

Table 22: cpe voip add Command Option Explanations

Command Option	Description
interface/port number	ONU port ID and POTS UNI port ID of the physical interfaces.
admin-state < up | down>	Activates or deactivates the functions performed by the POTS port for this subscriber. Default is up.
dial-number < 36 byte character string >	Specifies the subscriber directory number. **Note:** The dial-number field is only used with SIP.

Table 22: cpe voip add Command Option Explanations

Command Option	Description
username < 25 byte unique character string >	Defines the customer id used for the display attribute in outgoing SIP messages. **Note:** The username field is only used with SIP.
password < 25 byte character string >	Contains the SIP user identification used for authentication. **Note:** The password field is only used with SIP
rx-gain < -12db - 6db >	Specifies a gain value for the received signal in the form of a 2s complement number. Valid values are -12 (-12.0 dB) to 6 (+6.0 dB).
tx-gain < -12db - 6db >	Specifies a gain value for the transmit signal in the form of a 2s complement number. Valid values are -12 (-12.0 dB) to 6 (+6.0 dB).
voip-server-profile <index | profile-name>	Associated cpe-voip-feature profile.
voip-features-profile <index | profile-name>	Associated cpe-voip-feature profile. If user specify profile index 1 or omit this field, a default profile is used. **Note:** The voip-features-profile field is only used with SIP.
voip-media-profile <index | profile-name>	Associated cpe-voip-media profile. If user specify profile index 1 or omit this field, a default profile is used.

To create a CPE VoIP subscriber profile on an ONU with the **cpe voip add** command:

1. Create SIP services on ONU 1/3/1 POTS 1, and associate VoIP server profile 1, the default VoIP features profile, and the default VoIP media profile with them.

   Make sure the POTS port matches the port ID assigned during the creation of the subscriber facing MXK-194/198 bridge and CPE connection.

   ```
   zSH> CPE> VOIP> add 1/3/1/1 dial-number 2012020013
   username 2012020013 password 123456
   voip-server-profile 1
   Service has been created
   ```

2. Create SIP services on ONU 1/3/1 POTS 2, and associate VoIP server profile 1, VoIP features profile featurelist1, and VoIP media profile T38fax with them.

```
zSH> CPE> VOIP> add 1/3/1/2 dial-number 2012020014
username 2012020014 password 123456
voip-server-profile 1 voip-features-profile
featurelist1 voip-media-profile T38fax
Service has been created
```

3 Show all the VoIP subscriber profiles.

```
zSH> CPE> VOIP> show all
         Port  Admin                  Voip-Server  Voip Features  Voip Media
   CPE     Number State Rx Gain Tx Gain Profile Index Profile Index Profile Index
========== ====== ===== ======= ======= ============= ============= =============
     1/3/1      1    up       0       0             1             1             1
Dial Number   : 2012020013
Username      : 2012020013
Password      : 123456

            Port  Admin                  Voip-Server  Voip Features  Voip Media
   CPE     Number State Rx Gain Tx Gain Profile Index Profile Index Profile Index
========== ====== ===== ======= ======= ============= ============= =============
     1/3/1      2    up       0       0             1             2             2
Dial Number   : 2012020014
Username      : 2012020014
Password      : 123456
2 services displayed
```

Testing the VoIP configuration

1 Connect an ONT downlink POTS port to a VoIP phone.

Make sure the POTS port number matches the one you assigned for voice service with the **bridge add** command. In this example, you can connect either POTS 1 or POTS 2 to the VoIP phone.

2 Pick up the phone, you should be able to hear the dial tone and be able to make and receive a phone call.

Creating PWE on Dynamic OMCI on uplink and downlink bridges

For Pseudo-Wire Emulation (PWE) service we recommend using uplink and downlink bridge configurations.

To create PWE service on the PWE (i.e. CES) ports on the same ONU, use the following steps:

- Creating a GPON traffic profile, page 404
- Creating Uplink/Downlink bridge and CPE connection, page 404
- Creating CPE IP server profile, page 405
- Creating a CPE IP profile for the PWE service and associate it with a CPE IP server profile, page 406

- Creating a CPE PWE profile, page 406
- Creating PWE service on ONU CES ports and associate it with a CPE PWE profile, page 408
- Testing the PWE configuration, page 409

Creating a GPON traffic profile

Add the GPON traffic profile.

The following GPON traffic profile is recommended for up to four PWE (i.e. CES) ports. Note that it is not recommended to use the DBA configuration fields in the GPON traffic profile for PWE service.

```
zSH> new gpon-traffic-profile 4

gpon-traffic-profile  4
Please provide the following: [q]uit.
guaranteed-upstream-bw: ->  {0}: 13312
traffic-class: ---------->  {ubr}: cbr
compensated: ------------>  {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..
```

Creating Uplink/Downlink bridge and CPE connection

This section will create an uplink bridge and a downlink bridge on the MXK-194/198 and create CPE connection on the ONU for PWE service:

1. Create an uplink bridge on the uplink interface.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 503 tagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record ethernet4-503/bridge
Bridge-path added successfully
```

2. Create a downlink bridge on the downlink interface and create a CPE connection for all the CES ports on the ONU.

   This example creates a PWE service for all CES ports on the ONU 1/3/1 and inserts VLAN 503 with COS 3 to the tagged PWE packets.

```
zSH> bridge add 1-1-3-1/gponONU gem 1350 gtp 4
downlink cos 3 vlan 503 tagged pwe
Adding bridge on 1-1-3-1/gponONU
```

```
Created bridge-interface-record
1-1-3-1350-gponport-503/bridge
```

3 On MXK-194/198, run the **bridge show** command to view the MXK-194/198 bridges.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-----------------------------------------------------------------------------------------------------------
dwn              Tagged  300    1/1/3/1/gpononu           1-1-3-710-gponport-300/bridge  UP  D 00:19:c7:0d:11:bf
dwn              Tagged  503    1/1/3/1/gpononu           1-1-3-1350-gponport-503/bridge UP
dwn              Tagged  999    1/1/3/1/gpononu           1-1-3-650-gponport-999/bridge  UP
dwn              Tagged  1001   1/1/3/1/gpononu           1-1-3-510-gponport-1001/bridge UP
upl              Tagged  300    1/1/4/0/eth               ethernet4-300/bridge           UP
upl              Tagged  503    1/1/4/0/eth               ethernet4-503/bridge           UP
upl              Tagged  999    1/1/4/0/eth               ethernet4-999/bridge           UP  S VLAN 999 default
upl              Tagged   1001  1/1/8/0/eth               ethernet8/bridge               UP  S VLAN 1001 default
8 Bridge Interfaces displayed
```

4 View the CPE connections.

```
zSH> bridge show onu
           GEM   ONU     ONU UNI    OLT           OLT
ONU        Port  UNI     VLAN/SLAN  VLAN/SLAN     MVR   Service  OLT Bridge                                ST
-----------------------------------------------------------------------------------------------------------
1/3/1      510   eth 1   1001/----  Tagged  1001        data     1-1-3-510-gponport-1001/bridge          UP
1/3/1      510   eth 2   1001/----  Tagged  1001        data     1-1-3-510-gponport-1001/bridge          UP
1/3/1      650   eth 3              Tagged  999         iptv     1-1-3-650-gponport-999/bridge           UP
1/3/1      710   pots               Tagged  300         sip      1-1-3-710-gponport-300/bridge           UP
1/3/1      1350  ces                Tagged  503         pwe      1-1-3-1350-gponport-503/bridge          UP
4 Bridge Interfaces displayed
5 GPON ONU Connections displayed
```

Creating CPE IP server profile

The default CPE IP server profile specified the DHCP as the host IP option. It indicates CPE will get the host IP address automatically from the DHCP server.

Note: CPE IP server profile can only be deleted when it is not associated by any CPE IP profiles.

The following example creates a static CPE IP server profile:

1 Create a CPE IP server profile with profile-name. The profile index will be generated automatically.

```
zSH> CPE> IP> SERVER> add ip-pwe-server host-ip-option static netmask
255.255.255.0 gateway 172.10.10.1 primary-dns 172.111.142.50 secondary-dns
172.112.142.50
Profile "ip-pwe-server" has been created with index 2
```

2 Show the default CPE IP server and user-created CPE IP servers.

```
zSH> CPE> IP> SERVER> show all
                                                host IP
   Index              Profile Name               option    netmask          gateway        primary DNS    secondary DNS
========== ================================== ====== =============== =============== =============== =============
1          Default_Cpe_Ip_Server                 dhcp   255.255.255.0         0.0.0.0         0.0.0.0         0.0.0.0
2          ip-pwe-server                       static   255.255.255.0     172.10.10.1  172.111.142.50  172.112.142.50
2 entries found.
```

Creating a CPE IP profile for the PWE service and associate it with a CPE IP server profile

Create a CPE IP profile for the PWE service and associate it with a CPE IP server. If there is no CPE IP server specified, the default CPE IP server with dhcp option will be used.

1. Create a CPE IP profile on ONU 1/3/1 for the PWE service, and associate it with an IP server. Note that the host-ip field must be specified if a static IP server is used.

```
zSH> CPE> IP> add 1/3/1 pwe host-ip 172.10.10.20 ip-server ip-pwe-server
Service has been created
```

2. Show all the CPE IP profiles.

```
zSH> CPE> IP> show all
   CPE     Service     host IP     IP Server
========== ======= =============== =========
     1/3/1    VOIP         0.0.0.0         1
     1/3/1     PWE    172.10.10.20         2
2 services displayed
```

Creating a CPE PWE profile

Create a CPE PWE profile for the common PWE service or use the default CPE PWE profile.

Note: CPE PWE profile can only be deleted when it is not associated by any CPE PWE subscriber profiles.

Command:

cpe pwe common add *<profile-name>*
[**line-type** < **ds1** | **e1** >]
[**encoding** < **b8zs** | **ami** | **hdb3** | **b3zs** >]
[**service-type** < **unstructured** | **octetalignedunstruct** | **structured** >]
[**timing-mode** < **network** | **differential** | **adaptive** | **loop** >]
[**payload-size** < *value* >]
[**jitter-buf-max** < *value* >]
[**jitter-buf-desired** < *value* >]

Table 23: cpe pwe common add Command Option Explanations

Command Option	Description
profile-name	Specifies a unique profile name for cpe pwe profile.
line-type < ds1 | e1>	Specifies the line type used. ds1 or e1. Default value is e1.

Table 23: cpe pwe common add Command Option Explanations

Command Option	Description
encoding < b8zs | ami | hdb3 | b3zs >	Specifies the line coding scheme. b8zs is used for ds1 line-type, hdb3 is used for e1 line-type. Default value is hdb3.
service-type < unstructured | octetalignedunstruct | structured >	Specifies the basic service type, either a transparent bit pipe or an encapsulation that recognizes the underlying structure of the payload. Default value is unstructured.
timing-mode < network | differential | adaptive | loop >	Selects the timing mode of the TDM service. If RTP is used. Default value is network.
payload-size < value >	Specifies the number of payload bytes per packets. Valid only if service-type is unstructured or octetalignedunstruct (unstructured octet aligned). Valid choices depend on the TDM service, but must include the following. Other choices are at the vendor's discretion. Values: **192** For DS1 service **200** For DS1 service, required only if service-type octetalignedunstruct is selected **256** For E1 service **1024** For DS3 and E3 service.
jitter-buf-max < value >	Specifies the desired maximum depth of the playout buffer in the PSN to TDM direction. The value is expressed as a multiple of the 125 microseconds frame rate. The value 0 selects the ONT's internal policy.
jitter-buf-desired < value >	Specifies the desired nominal fill depth of the playout buffer in the PSN to TDM direction. The value is expressed as a multiple of the 125 microseconds frame rate. The value 0 selects the ONT's internal policy.

1 Add a PWE profile. If the user-end device is PWE T1 device, line-type must be ds1, and encoding might be changed as well.

```
zSH> CPE> PWE> COMMON> add pwe-t1 line-type ds1 encoding b8zs
Profile "pwe-t1" has been created with index 2
```

If the user-end device is PWE e1 device, you can use the default value of the line-type and encoding, which are e1 and hdb3.

```
zSH> CPE> PWE> COMMON> add pwe-e1
```

2 Show all the PWE profiles.

```
zSH> CPE> PWE> COMMON> show all
                                                   line                                    payload jitter JBuf
   Index              Profile Name                 type  encoding Service type  timing Mode size    BufMax Desired
========== ==================================== ===== ======== ============ ============ ======= ====== =======
1          Default_Cpe_Pwe                      e1    hdb3     unstructured adaptive     250     128    64
2          pwe-t1                               ds1   b8zs     unstructured adaptive     250     128    64
2 entries found.
```

Creating PWE service on ONU CES ports and associate it with a CPE PWE profile

Create a CPE PWE subscriber profile on ONU CES ports *manually* and associate it with a CPE PWE profile by using the **cpe pwe add** command. If there is no CPE PWE profile specified, the default CPE PWE profile will be used.

Note that a CPE PWE subscriber profile will also be created *automatically* if users set the ONU UNI CES port's admin-state with the **gpononu port** command. For the details, refer to *Administration of subscriber facing ports* on page 455.

After a CPE PWE subscriber profile is created, to change the settings in that profile, you can use the **cpe pwe modify** command, which has the same command syntax as the **cpe pwe add** command.

Command:

```
cpe pwe add <interface>/<port number>
        [ admin-state < up | down > ]
        [ near-end-port < value > ]
        [ far-end-ip < IP Address > ]
        [ far-end-port < value > ]
        [ line-length < value > ]
        [ pwe-profile < index | profile-name > ]
```

Table 24: cpe pwe add Command Option Explanations

Command Option	Description
interface/port number	ONU port ID and CES UNI port ID of the physical interfaces.
admin-state *value*	Activates or deactivates the functions performed by the CES port for this subscriber. Possible values are up, down. Default value is up.
near-end-port *value*	When the pseudowire service is transported via IP, this attribute specifies the port number of the near-end TCP/UDP service. Default is 57000 + port number.
far-end-ip *value*	When the pseudowire service is transported via IP, this attribute specifies the IP address or resolved name of the far-end termination point.
far-end-port *value*	When the pseudowire service is transported via IP, this attribute specifies the port number of the far-end TCP/UDP service. Default is 57000 + port number.
line-length *value*	Specifies the length of the twisted pair cable from a DS1 physical UNI to the DSX-1 cross-connect point. In the unit of feet. Default is 0.
pwe-profile *index* | *profile-name*	Points to the associated CPE PWE profile. If this field is not specified or is 1, the default CPE PWE profile (index 1) will be used.

To create a CPE PWE subscriber profile with the **cpe pwe add** command:

1. Create a CPE PWE subscriber profile on ONU 1/3/1 CES port 1 and associate with a CPE PWE profile.

   Make sure the CES port matches the port ID assigned during the creation of the subscriber facing MXK-194/198 bridge and CPE connection.

```
zSH> CPE> PWE> add 1/3/1/1 near-end-port 57001 far-end-ip 10.10.10.1 far-end-port
57001 pwe-profile pwe-t1
Service has been created
```

2. Create another CPE PWE subscriber profile on ONU 1/3/1 CES port 2 and associate with a CPE PWE profile.

```
zSH> CPE> PWE> add 1/3/1/2 near-end-port 57002 far-end-ip 10.10.10.1 far-end-port
57002 pwe-profile pwe-t1
Service has been created
```

3. Show all the CPE PWE subscriber profiles

```
zSH> CPE> PWE> show 1/3/1

   CPE     Port Number   Admin State Near End Port    Far End Ip   Far End Port Line Length Pwe Profile

========== ============= =========== ============= =============== ============ =========== ===========

     1/3/1            1          up         57001      10.10.10.1        57001           0           2

     1/3/1            2          up         57002      10.10.10.1        57002           0           2

2 services displayed
```

Testing the PWE configuration

1. Connect an ONT downlink CES port to a T1/E1 device.

   Make sure the line type of the device matches the configuration you specified. In this example, you can use a T1 device.

   Make sure the CES port number matches the one you assigned for PWE service. In this example, you can connect CES 1 or CES 2 to the T1 device.

2. Verify if the T1/E1 device can get PWE service.

   We are using a T1/E1 device has a Sync light. If the Sync light (green) is on, it means the T1/E1 device got service.

Creating RF on Dynamic OMCI

We are going to create RF service on another ONU with ONU model ZNID-GPON-2511. Same as ZNID-GPON-5114, before creating service on this ONU, we have to make sure the internal ME profile of this ONU is specified in the **onu set** command. RF service is different than other services, there is no need to create bridge configurations for it.

To create RF service on the RF ports on an ONU, use the following steps:

- Specifying internal ME profile and activating the ONU, page 410
- Creating RF service on ONU RF ports, page 410

- Testing the RF configuration, page 411

Specifying internal ME profile and activating the ONU

Based on the ONU model, assign the matching internal ME profile to this ONU. It is a way to indicate this ONU is provisioned by dynamic OMCI.

1. Specify an internal ME profile for an ONU.

   After specifying internal ME profile zhone-2511 for ONU 1/3/2, the dynamic OMCI supports are associated with the ONT.

   ```
   zSH> onu set 1/3/2 meprof zhone-2511
   ```

2. To activate an ONT first run the **gpononu show** command to display the ONTs currently on the OLT, and discover the available serial numbers.

   The **gpononu show** command has options to select by slot and OLT. If you run the command without defining the slot/OLT the command will check for ONTs on every port of every card and depending on the number of cards, may take a long time to complete.

```
zSH> gpononu show 1/3
Free ONUs for slot 1 olt 3:
    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 1 olt 3:
sernoID   Vendor  Serial Number     sernoID   Vendor  Serial Number
    2       ZNTS    56725008
```

3. Run the **gpononu set** command to associate a serial number to the appropriate ONT:

   ```
   zSH> gpononu set 1/3/2 2
   Onu 2 successfully enabled with serial number ZNTS
   56725008
   ```

Creating RF service on ONU RF ports

To create a CPE RF subscriber profile on ONU RF ports *manually*, use the **cpe rf add** command.

Note that a CPE RF subscriber profile will be created *automatically* if users set the ONU RF port's admin-state with the **gpononu port** command. For the details, refer to *Administration of subscriber facing ports* on page 455.

After a CPE RF subscriber profile is created, to change the settings in that profile, you can use the **cpe rf modify** command, which has the same command syntax as the **cpe rf add** command.

Command:

cpe rf add *<interface>/<port number>*

[**admin-state** < **up** | **down** >]

To create a CPE RF subscriber profile with the **cpe rf add** command:

1. Create an RF service on ONU 1/3/2 RF port 1. If there is no RF port provided then the next available port number will be chosen.

```
zSH> CPE> RF > add 1/3/2/1
```

2. Show all the RF subscriber profile.

```
zSH> CPE> RF> show all
   CPE     Port Number  Admin State
========== =========== ============
    1/3/2             1           up
1 services displayed
```

The default value of the Admin state is up.

3. Show the RF service on CPE level.

```
zSH> CPE> show 1/3/2
CPE 1/3/2
Service: RF
Port Admin
---- -----
   1    up
```

Testing the RF configuration

1. Connect an ONT downlink RF port to a TV through the RF cable.

   Make sure the RF port number matches the one you assigned for RF service. In this example, you can connect RF 1 to the TV.

2. Check the TV signal.

View all services on an ONU

To view all services on an ONU, use the **cpe show** slot/olt/onu command.

This example shows ONU 1/3/1 has configured with triple-play services and plus PWE service.

```
zSH> CPE> show 1/3/1

CPE 1/3/1
Service: DATA
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN Admin Oper
---- ------ ------------- ------------- ----- -----
 510  eth 1     1001/----   Tagged 1001    up  down
 510  eth 2     1001/----   Tagged 1001    up  down

Service: IPTV
```

```
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN Admin Oper  Video Prof
---- ------ ------------- ------------- ----- ----- ----------
 650  eth 3                  Tagged 999    up  down          1

Service: SIP
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN    Host IP      IP Srvr Prof
---- ------ ------------- ------------- --------------- ------------
 710   pots                  Tagged 300                            1
    Port Admin Oper  Srvr Prof Feature Prof Media Prof DN             User Name
Password
     ==== ===== ===== ========= ============ ========== ===============
=============== ===============
      1   up    up       1          1        1     2012020013     2012020013
123456
      2   up    up       1          2        2     2012020014     2012020014
123456

Service: PWE
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN     Host IP     IP Srvr Prof
---- ------ ------------- ------------- --------------- ------------
1350    ces                  Tagged 503   172.10.10.20            2
     Port Admin Oper  Near End Port   Far End Ip    Far End Port Pwe Profile
     ==== ===== ===== ============= =============== ============ ===========
        1    up  down         57001      10.10.10.1        57001           2
        2    up  down         57002      10.10.10.1        57002           2
```

Delete CPE subscriber profiles on an ONU

Deleting CPE subscriber profiles on an ONU

The **cpe delete** *slot*[/*olt*[/*onu*]] command deletes all the CPE subscriber profiles that were created on ONUs.

If you want to delete all the ONU configuration on an ONU, use the **onu delete** *slot*[/*olt*[/*onu*]] command. For the details, refer to *Deleting ONU configuration on an ONU* on page 470.

1 Removes all CPE subscriber profiles on ONU 1/3/1.

```
zSH> cpe delete 1/3/1
Ok to delete ALL CPE profiles for ONU 1/3/1? [yes] or [no]: yes
Do you want to exit from this request? [yes] or [no]: no
Are you sure? [yes] or [no]: yes
Operation completed
```

2 Verify the CPE subscriber profiles are removed on this ONU.

```
zSH> CPE> ETH> show 1/3/1
No services found.

zSH> CPE> VOIP> show 1/3/1
No services found.

zSH> CPE> PWE> show 1/3/1
No services found.
```

3 Verify all settings that were specified in the CPE subscriber profiles are removed on this ONU.

```
zSH> CPE> show 1/3/1

CPE 1/3/1
Service: DATA
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN Admin Oper
---- ------ ------------- ------------- ----- -----
 510  eth 1     1001/----     Tagged 1001       down
 510  eth 2    1001/----     Tagged 1001      down

Service: IPTV
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN Admin Oper  Video Prof
---- ------ ------------- ------------- ----- ----- ----------
 650  eth 3                 Tagged 999         down

Service: SIP
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN    Host IP      IP Srvr Prof
---- ------ ------------- ------------- --------------- ------------
 710   pots                 Tagged 300

Service: PWE
GEM   UNI   UNI VLAN/SLAN OLT VLAN/SLAN     Host IP     IP Srvr Prof
---- ------ ------------- ------------- --------------- ------------
1350    ces                 Tagged 503
```

Move ONU configuration

The ONU move feature allows user to move the ONU configuration from one ONU to another ONU, or from one OLT port to another OLT port. The ONU configuration here includes all the CPE subscriber profiles, CPE connections, GEM ports, bridges, and assigned serial number on the ONU. This feature could be used in many cases. For example, if the OLT SFP has some hardware failures, you can just simply unplug the fiber from this OLT port, plug-in to another OLT port, and move the ONU configuration over to the new OLT port.

Note: The destination ONU must have no ONU configuration on it.

Note: After moving ONU configuration to the destination ONU, the ONU configuration on the source ONU will be deleted.

When moving ONUs by OLT port , the ONU's GEM port IDs will be preserved. When moving a single ONU, the ONU may be assigned a different GEM port ID when it is moved, because another ONU may have already allocated the GEM port ID.

To move the ONU configuration from a source ONU to the destination ONU, use the **gpononu move** commands.

Command:

gpononu move <slot>[/<olt>[/<onu>]]**to** <slot>[/<olt>[/<onu>]]

To move ONU configuration from one individual ONU to another:

zSH> onu move 1/1/1 to 1/2/4

To move ONU configuration of all ONUs under one OLT port to another:

zSH> onu move 1/1 to 1/2

Moving ONU configuration from one OLT to another OLT

The following example shows how to move the ONU configuration from one OLT port to another:

Move the ONU configuration from all ONUs under OLT port 1/1 to all ONUs under OLT port 1/2.

```
zSH> onu move 1/1 to 1/2
Ok to move ONUs from OLT 1/1 to OLT 1/2? [yes] or [no]:
yes
Do you want to exit from this request? [yes] or [no]:
no
Are you sure? [yes] or [no]: yes

querying ONU 1/2/1
querying ONU 1/2/64
querying ONU 1/2/63
querying ONU 1/2/62
querying ONU 1/2/61
querying ONU 1/2/60
    :      :
querying ONU 1/2/4
querying ONU 1/2/3
querying ONU 1/2/2
copying  ONU 1/1/1 to 1/2/1
copying  ONU 1/1/64 to 1/2/64
copying  ONU 1/1/63 to 1/2/63
copying  ONU 1/1/62 to 1/2/62
    :    :
copying  ONU 1/1/5 to 1/2/5
copying  ONU 1/1/4 to 1/2/4
copying  ONU 1/1/3 to 1/2/3
copying  ONU 1/1/2 to 1/2/2
deleting ONU 1/1/1
deleting ONU 1/1/64
deleting ONU 1/1/63
deleting ONU 1/1/62
     :     :
deleting ONU 1/1/5
deleting ONU 1/1/4
deleting ONU 1/1/3
deleting ONU 1/1/2
```

```
ONUs from OLT 1/1 have been moved to OLT 1/2
```

VLAN translation on ONU

For GPON line card, VLAN translation is performed on the ONU instead of on the MXK-194/198. VLAN translation on ONU is also referred to as CPE VLAN translation in this section.

In situations when devices in the core network expect unique identifiers for each subscriber, and because subscriber configurations on the MXK-194/198 can include large numbers of subscribers with pre-configured VLAN IDs, the ONU supports VLAN translation from the subscriber to the MXK-194/198 for VLANs sent to the core network.

When configuring bridges for VLAN translation, all network facing Ethernet ports on the MXK-194/198 must be tagged, all bridges facing the subscriber's ONU must be tagged, and all subscriber facing Ethernet UNI ports on the ONU must be tagged as well. Bridges that are untagged do not support translation. For VLAN translation to work, there must be a VLAN in the Ethernet packet when it arrives at the ONU from the subscriber facing Ethernet UNI port.

The range for translated VLAN IDs is 1-4090 (some VLANs are reserved).

Possible bridging configuration behaviors for CPE VLAN translation

Possible bridging configuration behaviors for CPE VLAN translation:

- the single-tagged Ethernet packet from subscriber facing ONU Ethernet UNI port, to subscriber facing single-tagged bridge, and to network facing single-tagged bridge with VLAN translation (*single-tagged to single-tagged*)

  Refer to *CPE VLAN translation on uplink and downlink bridges* on page 416.

bridge show onu command for CPE VLAN translation

The **bridge show onu** command displays both subscriber facing original VLAN IDs and the translated network facing VLAN IDs on ONU.

Figure 44: sample bridge show onu command display

```
zSH> bridge show onu
```

Original VLAN/SLAN at subscriber facing UNI port on ONU

VLAN/SLAN at network facing port on ONU

Tag mode: ST or Tagged

```
           GEM   ONU     ONU UNI    OLT           OLT
ONU        Port  UNI     VLAN/SLAN  VLAN/SLAN     MVR   Service  OLT Bridge                                ST
-------------------------------------------------------------------------------------------------------------------
1/3/1      510   eth 1   1001/----  Tagged  1001        data     1-1-3-510-gponport-1001/bridge            UP
1/3/1      510   eth 2   1001/----  Tagged  1001        data     1-1-3-510-gponport-1001/bridge            UP
1/3/1      650   eth 3              Tagged  999         iptv     1-1-3-650-gponport-999/bridge             UP
1/3/1      710   pots               Tagged  300         sip      1-1-3-710-gponport-300/bridge             UP
1/3/1      1350  ces                Tagged  503         pwe      1-1-3-1350-gponport-503/bridge            UP
```

Subscriber facing UNI port on ONU
eth
pots
ces

Type of services created on MXK bridge interface
data
iptv
sip
sipplar
h248
pwe

Name of MXK bridge interface

Status of CPE connection (incl. ONU status, GEM port status, and UNI port status)
UP = CPE connection can pass traffic
Down = CPE connection is not ready to pass traffic (not ready or has been administratively been switched to down)
ADN = At least one of the interfaces in the CPE connection has admin state as “down”.
TST = At least one of the interfaces in the CPE connection has admin state as “testing”.

```
4 Bridge Interfaces displayed
5 GPON ONU Connections displayed
```

CPE VLAN translation on uplink and downlink bridges

This section describes configuring uplink/downlink bridges on the MXK-194/198 and configuring CPE connections on the ONU for basic VLAN translation.

When configuring the ONU for VLAN translation, you *must* designate both the uplink bridge and the downlink bridge on the MXK-194/198 as *tagged*, and the Ethernet UNI port on the ONU as *tagged* as well. And the translated VLAN ID must be same at both the uplink bridge and the downlink bridge on the MXK-194/198. This allows the original VLAN ID on the subscriber side of the ONU to pass down to the subscriber, and the translated VLAN ID on the network side of the ONU to pass to the MXK-194/198 and the core network.

As shown in Figure 45, the VLAN ID 100 on subscriber facing Ethernet UNI port are translated on the ONU to VLAN ID 1001 for the subscriber facing downlink bridge and the network facing uplink bridge on the MXK-194/198.

Figure 45: Asymmetric bridges with CPE VLAN translation

Configuring single-tagged to single-tagged asymmetric bridges for CPE VLAN translation

1 Create a *tagged* uplink bridge with VLAN ID on the network facing Ethernet port.

```
zSH> bridge add 1-1-4-0/eth uplink vlan 1001 tagged
Adding bridge on 1-1-4-0/eth
Created bridge-interface-record ethernet4-1001/bridge
Bridge-path added successfully
```

Verify the bridge.

```
zSH> bridge show
     Orig
Type VLAN/SLAN     VLAN/SLAN  Bridge                          St  Table Data
-------------------------------------------------------------------------------
upl              Tagged 1001 ethernet4-1001/bridge          UP S VLAN 1001 default
1 bridges displayed
```

2 Create *tagged* downlink bridges on the MXK-194/198 with the translated VLAN ID, and create CPE connections with UNI-VLAN ID on subscriber facing Ethernet UNI ports.

This example translates uni-vlan 100 to vlan 1001.

```
zSH> bridge add 1-1-3-1/gpononu gem 710 gtp 1 downlink vlan 1001 tagged eth 1 uni-vlan 100
Adding bridge on 1-1-3-1/gpononu
Created bridge-interface-record 1-1-3-710-gponport-1001/bridge

zSH> bridge add 1-1-3-2/gpononu gem 720 gtp 1 downlink vlan 1001 tagged eth 1 uni-vlan 100
Adding bridge on 1-1-3-2/gpononu
Created bridge-interface-record 1-1-3-720-gponport-1001/bridge
```

Verify the downlink bridges. The **bridge show** command displays the VLAN ID the ONU translated.

```
zSH> bridge show vlan 1001
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-------------------------------------------------------------------------------------------------------------
dwn             Tagged  1001   1/1/3/2/gpononu           1-1-3-720-gponport-1001/bridge UP
dwn             Tagged  1001   1/1/3/1/gpononu           1-1-3-710-gponport-1001/bridge UP
upl             Tagged  1001   1/1/4/0/eth               ethernet4-1001/bridge          UP  S VLAN 1001 default
3 Bridge Interfaces displayed
```

Verify the CPE connections. The **bridge show onu** command displays the original VLAN ID of the ONU (i.e. under the column of ONU UNI VLAN/SLAN) and the VLAN ID the ONU translated (i.e. under the column of OLT VLAN/SLAN).

```
zSH> bridge show onu vlan 1001
```

```
           GEM   ONU     ONU UNI    OLT           OLT
ONU        Port  UNI     VLAN/SLAN  VLAN/SLAN     MVR   Service  OLT Bridge                                ST
-----------------------------------------------------------------------------------------------------------------
1/3/1      710   eth 1    100/----  Tagged  1001        data     1-1-3-710-gponport-1001/bridge          UP
1/3/2       720   eth 1     100/---- Tagged  1001      data     1-1-3-720-gponport-1001/bridge          UP
2 Bridge Interfaces displayed
2 GPON ONU Connections displayed
```

Deleting single-tagged to single-tagged asymmetric bridges with CPE VLAN translation

1 View the existing bridges and CPE connections.

```
zSH> bridge show vlan 1001
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                          St  Table Data
-----------------------------------------------------------------------------------------------------------------
dwn             Tagged  1001   1/1/3/2/gpononu           1-1-3-720-gponport-1001/bridge UP
dwn             Tagged  1001   1/1/3/1/gpononu           1-1-3-710-gponport-1001/bridge UP
upl             Tagged  1001   1/1/4/0/eth               ethernet4-1001/bridge          UP  S VLAN 1001 default
3 Bridge Interfaces displayed

zSH> bridge show onu vlan 1001
           GEM   ONU     ONU UNI    OLT           OLT
ONU        Port  UNI     VLAN/SLAN  VLAN/SLAN     MVR   Service  OLT Bridge                                ST
-----------------------------------------------------------------------------------------------------------------
1/3/1      710   eth 1    100/----  Tagged  1001        data     1-1-3-710-gponport-1001/bridge          UP
1/3/2       720   eth 1   100/----  Tagged  1001        data     1-1-3-720-gponport-1001/bridge          UP
2 Bridge Interfaces displayed
2 GPON ONU Connections displayed
```

2 Delete the downlink bridges and stacked CPE connections. CPE VLAN ID translation uses the ethernet port ID and original VLAN ID in the **bridge delete** syntax.

```
zSH> bridge delete 1-1-3-710-gponport-1001/bridge eth 1 uni-vlan 100
GPON ONU Connection 1-1-3-710-gponport-1001/bridge/1/1/100/0 has been deleted
1-1-3-710-gponport-1001/bridge delete complete

zSH> bridge delete 1-1-3-720-gponport-1001/bridge eth 1 uni-vlan 100
GPON ONU Connection 1-1-3-720-gponport-1001/bridge/1/1/100/0 has been deleted
1-1-3-720-gponport-1001/bridge delete complete
```

3 Delete the uplink bridge. Uses the translated VLAN ID in the **bridge delete** syntax.

```
zSH> bridge delete ethernet4-1001/bridge vlan 1001
Bridge-path deleted successfully
ethernet4-1001/bridge delete complete
```

Browser based GPON zNID installation

This section includes the following topics:

- Configuring a bridge for data, GPON, page 419
- Configuring IPTV, GPON, page 428
- Configuring VoIP, GPON, page 430
- Configuring Triple Play: Data, Video and Voice, GPON, page 435

Figure 46: Installation procedure for browser based zNIDs

Configuring a bridge for data, GPON

To configure a bridge to the zNID, you must have a bridge on the MXK-194/198 (The MXK-194/198 acts as the OLT; in fact, each port can be considered a separate OLT). To build a bridge that reaches the subscriber devices bridges need to be built on the zNID.

For each service we will be adding a separate bridge with its own VLAN. For the data and video services we will set up an uplink and downlink bridge. From the perspective of each access device, the MXK-194/198 and zNID, this

means creating a bridge from the upstream interface to the downstream interface.

For data services we will create a bridge on the MXK-194/198 from the Internet uplink to the downlink. On the zNID we will create a bridge from the WAN PON interface to LAN 2 port. For data coming from the PC to the LAN port we will need to add a VLAN header, so the packets will be designated VLAN 200. Packets going downstream to the PC on LAN 2 port will likewise have the VLAN information stripped because the PC does not need tagged packets; only packets with VLAN 200 will be delivered to the LAN 2 interface.

Figure 47: Bridges on the MXK-194/198 and zNID to pass data traffic

Clearing off the default settings of the zNID

We will remove the default connections, so we can go through the steps of creating a solution.

1. Click **Network Connections** in the left hand menu pane
2. In the **Network Connections** page, delete **Data VLAN 200** by clicking the delete action icon for that item, then click **OK** to confirm
3. Delete **Management VLAN 300** by clicking the delete action for that item, then click **OK** to confirm

Creating a bridge on the zNID

We will create a bridge on the zNID.

1 Add a VLAN ID on the WAN interface

a Click **Network Connections** in the left hand menu pane

b In the **Networks Connection** page, click **New Connection**

c In the **Connection Wizard** screen select **Advanced Connection** and click **Next**

d In the **Advanced Configuration** screen select **VLAN Interface** and click **Next** near the bottom of the screen

e From the **Underlying Device** drop down select the physical WAN port (**WAN PON**) to associate with the VLAN ID

f In the **VLAN ID** text box enter the VLAN ID (200), then click **Next**

g In the **Connection Summary** screen select **Edit the Newly Created Connection**, then click **Finish**

h Verify the WAN Ethernet interface has been created, then click **OK**

Figure 48: The create WAN Ethernet interface

i Name the interface by clicking the edit icon for the **WAN Ethernet** interface you just created, then enter an appropriate name in the **Name** text box and click **OK**

We will use **Data VLAN 200 WAN Ethernet**.

2 Add a VLAN ID to the LAN switch

a Click **Network Connections** in the left hand menu pane

b In the **Networks Connection** page, click **New Connection**

c In the **Connection Wizard** screen select **Advanced Connection** and click **Next**

d In the **Advanced Connection** screen select **VLAN Interface** and click **Next**

e From the **Underlying Device** dropdown menu select the Ethernet switch to associate with the VLAN ID (**LAN Hardware Ethernet Switch**)

f In the **VLAN ID** text box enter the VLAN ID (200), then click **Next**

g In the **Connection Summary** screen select **Edit the Newly Created Connection**, then click **Finish,** view the screen then click **OK**

h Name the interface by clicking the edit icon for the **LAN Ethernet** interface you just created, then enter an appropriate name in the **Name** text box and click **Next**

We will use **Data VLAN 200 LAN Ethernet**.

3 Build the bridge between (among) the interfaces

- **a** Click **Network Connections** in the left hand menu pane
- **b** In the **Networks Connection** page, click **New Connection**
- **c** In the **Connection Wizard** screen select **Advanced Connection** and click **Next**
- **d** In the **Connection Wizard** screen select **Network Bridging** then click **Next**

e Select the connections to combine in the bridge (Data VLAN 200 WAN Ethernet and Data VLAN 200 LAN Ethernet) then click **Next**

f In the **Connection Summary** screen select **Edit the Newly Created Connection** then click **Finish**

g In the **Internet Protocol** drop down you should have **Obtain an IP address automatically** selected; click **OK**

h Rename the bridge by clicking the edit icon for the bridge you just created in the **Network Connections** screen, then enter **Data VLAN 200 Bridge** in the **Name** text box and click **OK**

4 Map the VLAN ID to the physical port

a Click **Network Connections** in the left hand menu pane

b In the **Network Connections** screen click on the **LAN Hardware Ethernet Switch** link

c At the bottom of the **LAN Hardware Ethernet Switch Properties** screen click **Set** near the bottom of the screen

d Near the bottom of the **Configure LAN Hardware Ethernet Switch** screen select the action button for the port (Ethernet Port 2) to associate the VLAN

e In the **Port 2 Settings** screen, click **New Entry**

f Enter the VLAN ID (200) in the **VLAN ID** text box then click **OK**

g Click **OK** again to confirm

Creating an uplink and downlink bridge on the MXK-194/198

We will create an uplink and downlink for VLAN 200.

1 Create uplink and add bridge-path

The switch which is upstream from our MXK-194/198 is providing a network on VLAN 200. The packets are tagged from the switch.The bridge-path add command defines this bridge interface as the uplink for the VLAN 200 downlinks.

Note: For all of the scenario examples in this application guide we are using the same uplinks/upstream interfaces, so if you have already created the uplink/upstream bridge interfaces, you will not need to recreate the data uplink/bridge-bridgepath here.

```
zSH> bridge add 1-1-5-0/eth uplink vlan 200 tagged
Adding bridge on 1-1-5-0/eth
Created bridge-interface-record ethernet5-200/bridge
Bridge-path added successfully
```

2 Add the downlink to the zNID

We are just passing the single tagged 200 packets down to the zNID. The zNID needs to know the VLAN ID to pass them through properly as well. We will configure the zNID for that as well.

```
DOC-MXK> bridge add 1-1-1-501/gponport gtp 1 downlink vlan 200 tagged

Adding bridge on 1-1-1-501/gponport
Created bridge-interface-record 1-1-1-501-gponport-200/bridge
```

If the GEM port already exists then the gtp parameter is not required. See *Creating GPON Traffic Profile* on page 55.

Activate the zNID

This first time through, we will activate the zNID, and will not need to activate it again. Actually we will not be able to activate it again once it is up.

We could have activated the zNID as soon as it was connected through the fiber to the MXK-194/198.

Activating the zNID is a matter of discovering the ID of the zNID using the **gpononu show** command, then activating the zNID using the **gpononu set** command.

1 Discover the open ONTs on the line using the **gpononu show** command

In the **gpononu show** command we could have limited it to the port (**gpononu show 1/1**) to reduce the amount of time the discovery will take. If there are a lot of optical distribution networks (ODNs) on the MXK-194/198, commands like **gpononu show** without further qualifiers will attempt to do discovery for all of them. The more ODNs it does discovery for, the longer it will take.

```
DOC-MXK> gpononu show 1
Processing list of 512
This command may take several minutes to complete.
Do you want to continue?  (yes or no) [no] y
Free ONUs for slot 1 olt 1:
   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 4 olt 1:
sernoID   Vendor  Serial Number     sernoID   Vendor  Serial Number
 5        ZNTS    466425
```

2 Set the discovered **sernoID** for the zNID

```
DOC-MXK> gpononu set 1/1/1 5
Onu 1 successfully enabled with serial number ZNTS 466425
```

Testing the data bridge

To test the connection, we will put the laptop on the LAN 2 port, ping to the Internet gateway and open a browser. Pinging to the Internet gateway proves the data path is open.

1 In the **Network Connections** screen you should see the **Status** as **Connected**

The view above is the advanced view (click the **Advanced** button).

2 Open a DOS window and ping the upstream gateway (provided in your environment setup)

If you cannot ping it means you do not have data access to your gateway. If you show connected on the WAN PON and the bridge, it means you have access on VLAN 200.

You should be able to verify the gateway is up by pinging from the MXK-194/198. On the MXK-194/198, just do a normal ping to the gateway as you would from a DOS window.

If you have access to the gateway from the MXK-194/198, do a few **bridge stats** commands to verify the bridge is accepting and receiving packets

3 Open a browser to a public site

As long as you can ping you are showing that you have a data path through the zNID and the MXK-194/198 to the Internet gateway. As long as that gateway has access to the Internet you should be able to open a browser and bring up a page.

Configuring IPTV, GPON

We will use the fast path feature to define a 999 VLAN which pushes the packets directly out to LAN port 3.

Figure 49: Passing data and video packets on separate bridges

Configuring the zNID for IPTV

1 Open Fast Path

   a From the left hand menu pane, click **Advanced**

   b Click the Fast Path icon

2 Select the ports

   a Select the subscriber port(s)

   We will select **Ethernet Port 3**

   b From the **WAN Device** dropdown menu, select the WAN interface (WAN PON)

3 Define the VLAN by entering the VLAN ID (999) in the **VID** text box

4 In the **Priority** text box enter a priority (3)

5 Click **OK**

Configuring the MXK-194/198 for IPTV

1 Create uplink and add bridge-path

The switch which is upstream from our MXK-194/198 is providing a video stream on VLAN 999. The packets are tagged from the switch.The bridge-path add command defines this bridge interface as the uplink for the VLAN 999 downlinks.

Note: For all of the scenario examples in this application guide we are using the same uplinks/upstream interfaces, so if you have already created the uplink/upstream bridge interfaces, you will not need to recreate the video uplink bridge-path here.

```
zSH> bridge add 1-1-5-0/eth uplink vlan 999 tagged
Adding bridge on 1-1-5-0/eth
Created bridge-interface-record ethernet5-200/bridge
Bridge-path added successfully
```

2 Add the downlink to the zNID

Just as with the data bridge, we will be passing single-tagged packets, this time vlan 999.

```
DOC-MXK> bridge add 1-1-1-901/gponport gtp 1 downlink vlan 999 tagged video
0/4
Adding bridge on 1-1-1-901/gponport
Created bridge-interface-record 1-1-1-901-gponport-999/bridge
```

3 Verify the bridge using the **bridge show** command

```
DOC-MXK> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN Physical         Bridge                     St Table Data
-------------------------------------------------------------------------------
tls          Tagged 160                    ipobridge-160/bridge   UP D 00:01:47:1a:fe:64
tls          Tagged160 1/1/1/1/gpononu  1-1-1-501-gponport-160/bridge  UP
tls                160 1/1/4/0/eth      ethernet4/bridge     UP D 00:00:86:43:3c:e4
                                                                 D 00:00:86:43:ec:69
upl          Tagged 999 1/1/5/0/eth     ethernet5-999/bridge UP S VLAN 999 default
dwn          Tagged 999 1/1/1/1/gpononu 1-1-1-901-gponport-999/bridge UP D
00:10:a4:b1:f0:bf
                                                                 D 01:00:5e:0a:0a:0a
dwn          Tagged 200 1/1/1/1/gpononu 1-1-1-501-gponport-200/bridge  UP  D
00:01:47:07:1d:fa
upl          Tagged 200 1/1/5/0/eth     ethernet5-200/bridge UP  S VLAN 200 default
```

Because the zNID is already active on the line it does not need to be activated.

Testing the IPTV bridge

Since we are using a laptop and software to emulate a set top box, we can ping out to the video server.

1. Ping the upstream gateway (provided in your environment setup)

   If you cannot ping it means you do not have data access to your gateway. If you show connected on the WAN PON and the bridge, it means you have access on VLAN 200.

   You should be able to verify the gateway is up by pinging from the MXK-194/198. On the MXK-194/198, just do a normal ping to the gateway as you would from a DOS window.

   If you have access to the gateway from the MXK-194/198, do a few **bridge stats** commands to verify the bridge is accepting and receiving packets

2. Open the STB emulation software and connect to the video server

   As long as you can ping you are showing that you have a data path through the zNID and the MXK-194/198 to the video server. You should be able to connect to the video stream with the STB emulation software.

Configuring VoIP, GPON

Use a TLS bridge instead an uplink and downlink bridge for VoIP. The TLS bridge allows bridge forwarding table timeouts, so if the MAC address has timed out, incoming calls from the softswitch will flood the TLS bridge and relearn the MAC address.

Figure 50: Adding a TLS bridge for voice services

Configuring the zNID for VoIP

1. Add a VLAN ID on the WAN interface as we did when building the data bridge

   a. Click **Network Connections** in the left hand menu pane

   b. In the **Networks Connection** page, click **New Connection**

   c. In the **Connection Wizard** screen select **Advanced Connection** and click **Next**

d In the **Advanced Configuration** screen select **VLAN Interface** and click **Next** near the bottom of the screen

e From the **Underlying Device** drop down select the physical WAN port (**WAN PON**) to associate with the VLAN ID

f In the **VLAN ID** text box enter the VLAN ID (300), then click **Next**

g In the **Connection Summary** screen select **Edit the Newly Created Connection**, then click **Finish**

h From the **Internet Protocol** drop down in the **Configure WAN Ethernet** screen (the interface you just created), select Obtain an IP Address Automatically.

From the DNS Server drop down you should have selected Obtain DNS Server Address Automatically

i Click **OK**

j Rename the interface by clicking the action icon for the **WAN Ethernet** interface you just created, entering **VoIP VLAN 300 Ethernet** in the **Name** text box, then click **OK**

2 Name the phone connection

a In the left hand menu pane, click **Voice Over IP**

b In the **Voice Over IP** screen, select the **Line Settings** tab

c In the **Line Settings** tab click the action button for the line text box enter the assigned phone number

d In the **User ID** text box enter the assigned User ID

e In the **Display Name text** box enter a name for the phone connection

3 Enter the authentication information

a In the **Authentication User Name** text box under **SIP Account** enter the assigned user name

b In the **Authentication Password** text box enter the assigned password

4 Select the phone connection type.

We will use SIP Proxy.

a Select Use **SIP Proxy**

b In the **Host Name or Address** text box under **SIP Proxy** enter the fully qualified address for the softswitch server.

c In the **Registrar Host Name or Address** text box under **SIP Proxy** enter the fully qualified address for the softswitch server.

d Click **OK**

Configuring the MXK-194/198 for VoIP

1 Add a TLS bridge interface on the uplink card

Note: For all of the scenario examples in this application guide we are using the same uplinks/upstream interfaces, so if you have already created the uplink/upstream bridge interfaces, you will not need to recreate the voice upstream link here.

```
DOC-MXK> bridge add 1-1-5-0/eth tls vlan 300 tagged
Adding bridge on 1-1-5-0/eth
Created bridge-interface-record ethernet5-300/bridge
```

2 Add a TLS bridge interface on the GPON line card

```
DOC-MXK> bridge add 1-1-1-701/gponport tls vlan 300 tagged
Adding bridge on 1-1-1-701/gponport
Created bridge-interface-record 1-1-1-701-gponport-300/bridge
```

3 Verify the bridge using the **bridge show** command

```
DOC-MXK> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN Physical         Bridge                St Table Data
-------------------------------------------------------------------------------------
tls         Tagged 160                 ipobridge-160/bridge UP  D 00:01:47:1a:fe:64
tls         Tagged 160 1/1/1/1/gpononu 1-1-1-501-gponport-160/bridge  UP
tls          160       1/1/4/0/eth     ethernet4/bridge     UP  D 00:00:86:43:3c:e4
                                                                 D 00:00:86:43:ec:69
upl         Tagged 999 1/1/5/0/eth     ethernet5-999/bridge UP  S VLAN 999 default
dwn         Tagged 999 1/1/1/1/gpononu 1-1-1-901-gponport-999/bridge UP D
00:10:a4:b1:f0:bf
                                                                 D 01:00:5e:0a:0a:0a
tls         Tagged 300 1/1/5/0/eth     ethernet5-300/bridge UP  D 00:00:86:43:3c:e4
                                                               D 00:00:86:43:ec:69
                                                                 D 00:01:47:1a:e4:74
                                                                 D 00:01:47:1a:fe:64
                                                                 D 08:00:20:b8:f6:58
tls         Tagged 300 1/1/1/1/gpononu 1-1-1-701-gponport-300/bridge  UP  D
00:01:47:07:1d:fa
dwn         Tagged 200 1/1/1/1/gpononu 1-1-1-501-gponport-200/bridge  UP  D
00:01:47:07:1d:fa
upl         Tagged 200 1/1/6/0/eth     ethernet6-200/bridge UP  S VLAN 200 default
```

Testing the VoIP configuration

1 When the VoIP connection accesses the softswitch it will show as registered on the Voice Over IP page

2 Making or receiving a phone call

Configuring Triple Play: Data, Video and Voice, GPON

If you have already followed the above procedures for configuring data, video and voice, then you should have triple play working for your solution.

- *Configuring a bridge for data, GPON* on page 419
- *Configuring IPTV, GPON* on page 428
- *Configuring VoIP, GPON* on page 430

GPON type B redundancy

The MXK-194/198 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 (i.e. GPON ports) on the MXK-194/198 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 51: GPON type B redundancy

With Zhone's GPON type B redundancy, a GPON redundancy group can have two GPON OLT ports. The ports can be on a 4 OLT ports MXK194 or a 8 OLT ports MXK198.

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-194/198 system, 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 or secondary port of the redundant pair, you will get an error message.

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
                                        Redundancy ---- Redundantcy Peer ----
 OLT        Interface          Status      State     OLT       Interface
===== ==================== ============ ========== ===== ====================
  1/1              1-1-1-0      OOS
  1/2              1-1-2-0      OOS
  1/3              1-1-3-0      OOS
  1/4              1-1-4-0      OOS
  1/5              1-1-5-0      UP
  1/6              1-1-6-0      UP
  1/7              1-1-7-0      OOS
  1/8              1-1-8-0      OOS
```

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 1-1-5-0/gponolt

The 1-1-5-0/gponolt is not part of any redundancy group
```

2 Add line redundancy

```
zSH> line-red add pri 1-1-5-0/gponolt sec 1-1-6-0/gponolt

This command may take several minutes to complete.

Do you want to continue?  (yes or no) [no] yes
Waiting for command completion...........

Protection pair has been created. 1-1-5-0/gponolt is primary and 1-1-6-0/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.

When the OLT port is ready, it will be displayed.

```
zSH> gponolt show redund
                                          Redundancy ---- Redundantcy Peer ----
 OLT        Interface          Status      State     OLT       Interface
===== ==================== ============ ========== ===== ====================
  1/1             1-1-1-0      OOS
  1/2             1-1-2-0      OOS
  1/3             1-1-3-0      OOS
  1/4             1-1-4-0      OOS
  1/5             1-1-5-0      UP      Primary     1/6             1-1-6-0
  1/6             1-1-6-0      UP      Secondary   1/5             1-1-5-0
  1/7             1-1-7-0      OOS
  1/8             1-1-8-0      OOS
```

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        1-1-5-0/gponolt               Active     UP
Secondary      1-1-6-0/gponolt               Standby    Trfc-Disable
```

4 Create interfaces on the GPON port

We will create a bridge interface:

```
zSH> bridge add 1-1-5-501/gponport gtp 1 downlink vlan 200 tagged
   Adding bridge on 1-1-5-501/gponport
   Created bridge-interface-record 1-1-5-501-gponport-200/bridge
```

Show the bridge:

```
zSH> bridge show
     Orig
Type VLAN/SLAN VLAN/SLAN   Physical         Bridge                St Table Data
-----------------------------------------------------------------------------------------
upl           Tagged 200 1/1/5/0/eth     ethernet5-200/bridge UP S VLAN 200 default
dwn           Tagged 200 1/1/5/1/gpononu 1-1-5-501-gponport-200/bridge UP D
00:00:86:43:3c:e4
```

5 Test line redundancy

a Bounce the port

```
zSH> port bounce 1-1-5-0/gponolt
1-1-5-0/gponolt set to admin state DOWN
1-1-5-0/gponolt set to admin state UP
```

b Show the line redundancy

```
zSH> line-red show 1-1-5-0
redundancy status for 1-1-5-0/gponolt:
         NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0
Interface-Type          Interface-Name        Oper-State Oper-Status
============== ============================== ========== ===========
Primary        1-1-5-0/gponolt               Standby    Trfc-Disable
Secondary      1-1-6-0/gponolt               Active     UP
```

c Show the bridge.

Notice that even though the OLT port 6 is now the active port the name of the bridge does not change (and makes it look like the bridge is coming from the OLT port 5, `1-1-5-501-gponport-200/bridge`).

```
zSH> bridge show

     Orig
Type VLAN/SLAN VLAN/SLAN Physical           Bridge              St Table Data
-------------------------------------------------------------------------------------
upl           Tagged 200 1/1/5/0/eth       ethernet5-200/bridge UP S VLAN 200 default
dwn           Tagged 200 1/1/5/1/gpononu 1-1-5-501-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 1-1-6-0
1-1-6-0 set to admin state DOWN
1-1-6-0 set to admin state UP
```

e Show the line redundancy

```
zSH> line-red show 1-1-6-0
redundancy status for 1-1-6-0/gponolt:
         NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0
Interface-Type          Interface-Name        Oper-State Oper-Status
============== ============================== ========== ===========
Primary        1-1-5-0/gponolt               Active     UP
Secondary      1-1-6-0/gponolt               Standby    Trfc-Disable
```

f Show the bridge

```
zSH> bridge show

     Orig
Type VLAN/SLAN VLAN/SLAN Physical           Bridge              St Table Data
-------------------------------------------------------------------------------------
upl           Tagged 200 1/1/5/0/eth       ethernet5-200/bridge UP S VLAN 200 default
```

```
dwn           Tagged 200 1/1/5/1/gpononu 1-1-5-501-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 1-1-5-0/gponolt
redundancy status for 1-1-5-0/gponolt:
         NOREBOOT standbytx ENABLE timeout 0 NONREVERTIVE revert timeout 0

Interface-Type          Interface-Name        Oper-State Oper-Status
============== ============================== ========== ===========
Primary        1-1-5-0/gponolt               Active     UP
Secondary      1-1-6-0/gponolt               Standby    Trfc-Disable
```

2 Remove the standby port from the redundancy group

```
zSH> line-red remove 1-1-6-0/gponolt
This command may take several minutes to complete.
Do you want to continue?  (yes or no) [no] yes
Waiting for command completion....
Interface 1-1-6-0/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 1-1-5-0/gponolt
The 1-1-5-0/gponolt is not part of any redundancy group
```

Switchover between active and standby GPON port

A switchover from active to standby GPON port can be done automatically or forced manually using the **port bounce** command.

A switchover from standby to active GPON port can be done manually by using the **line-red switch** 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-194/198 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-194/198, 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 from active to standby

A manual switchover from active to standby GPON port can be done by operator by using the **port bounce** *interface/physical interface type* command on an active GPON port.

```
zSH> port bounce 1-1-5-0/gponolt
1-1-4-0/gponolt set to admin state DOWN
1-1-4-0/gponolt set to admin state UP
```

Manually switched from standby to active

A manual switchover from standby to active can be done by operator by using the **line-red switch** *interface/physical interface type* command on a standby port:

1) Show the Oper-State

```
zSH>  line-red show 1-1-5-0/gponolt
> Interface-Type          Interface-Name        Oper-State Oper-Status
> ============== ============================== ========== ============
> Primary        1-1-5-0/gponolt               Active     UP
> Secondary      1-1-6-0/gponolt               Standby    Trfc-Disable
```

2) Switch the standby GPON port to active

```
zSH> line red switch 1-1-6-0/gponolt
1-1-6-0/gponolt is now active and 1-1-5-0/gponolt is
standby.
```

3) Check the Oper-State

```
zSH>  line-red show 1-1-5-0/gponolt
Interface-Type          Interface-Name        Oper-State Oper-Status
> ============== ============================== ========== ============
> Primary        1-1-5-0/gponolt                Standby    Trfc-Disable
> Secondary      1-1-6-0/gponolt                Active     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.
- 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.
- 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-194/198 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.

If 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, the GPON ONU serial number is 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 1/1
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 1/1 -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 1/1 -d
Processing list of 64
This command may take several minutes to complete.
Do you want to continue?  (yes or no) [no] yes
                                                Serial
   ONU       Name        Enabled     Model #      Number         OMCI files and
                                                                   profiles
=== ================= ======= =============== =============== ================
  1           1-1-1-1   Yes         2510       ZNTS 1341       ME   2210-me
                                                               GEN  2210-gen
```

```
  2           1-1-1-2   Yes                    ZNTS 1405       ME   2210-me
                                                               GEN  2210-gen
  3           1-1-1-3   Yes                    ZNTS 1263       ME   2210-me
                                                               GEN  2210-gen
  4           1-1-1-4   Yes                    ZNTS 1359       ME   2210-me
                                                               GEN  2210-gen
  5           1-1-1-5   Yes                    ZNTS 1285       ME   2210-me
                                                               GEN  2210-gen
  6           1-1-1-6   Yes                    ZNTS 1387       ME   2210-me
                                                               GEN  2210-gen
  7           1-1-1-7   Yes                    ZNTS 1335       ME   2210-me
                                                               GEN  2210-gen
  8           1-1-1-8   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.

Note: Attempting to provision an ONU that has a vendor ID other than ZNTS will not complete successfully and the system will return an error message, "`Alert!! Foreign ONT detected - cannot activate.`". Zhone requires official ONT interop for third party ONTs.

Associate a vendor ID and a hex serial number with an ONU and enable this ONU:

```
zSH> gpononu set 1/1/2 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 1/1 -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 1 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 1/1/3 vendorid ZNTS serno 2216690777
Onu 3 successfully enabled with serial number ZNTS 2216690777
```

GPON extended reach

The MXK-194/198 GPON solution supports extended reach. The maximum distance between the MXK-194/198 and the farthest ONT/ONU is 60 Km. The requirement when deploying ONTs are the maximum distance between two ONTs cannot exceed 20Km.

A key point to keep in mind when deploying GPON is optical budget. This becomes more critical when the distance between MXK-194/198 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-194/198 and farthest ONT: 40 Km
- Minimum Distance between MXK-194/198 and closest ONT to MXK-194/198: 20 Km

- Maximum distance between any two ONTs: 20 Km (Note this is always a constant)

50 Km:

- Maximum Distance between MXK-194/198 and farthest ONT: 50 Km
- Minimum Distance between MXK-194/198 and closest ONT to MXK-194/198: 30 Km
- Maximum distance between any two ONTs: 20 Km (Note this is always a constant)

60 Km:

- Maximum Distance between MXK-194/198 and farthest ONT: 60 Km
- Minimum Distance between MXK-194/198 and closest ONT to MXK-194/198: 40 Km
- Maximum distance between any two ONTs: 20 Km (Note this is always a constant)

Command to measure the distance between MXK-194/198 and ONT

To measure the approximate distance between MXK-194/198 and ONU, use the **onu status** command. The distance field shows the measurement in km.

```
zSH> gpononu status 1/1/1

                          Omci       Gpon    Download    OLT      ONT     Distance
ID    Onu   OperStatus ConfigState OnuStatus  State    Rx Power  Rx Power   (KM)
== ======== ========== =========== ========= ========= ========= ========= =======
1  1-1-1-1     Up     Done         Active   NoUpgrade -23.8 dBm -23.0 dBm   18
```

Commands to enable extended reach

By default, the MXK-194/198 supports a maximum distance of 20 Km between MXK-194/198 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-194/198.

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 both OLT and ONU side.

The user can view the upstream optical power level received at the OLT, and the downstream optical power level received at the ONU.

Note that the downstream optical power level received at the ONU (i.e. ONT Rx Power or ONT Receive Power) should be -28 or above for SFP-B+.

By default, if the measured upstream optical power of the ONU received at the OLT (i.e. OLT Rx Power or OLT Receive Power) is beyond the value range of -10 dBm to -30 dBm, the MXK-194/198 will trigger a local alarm, and send trap to ZMS. For the detail, refer to *GPON High and Low Receive Power Threshold Alarms* on page 499.

1 The following example shows the ONT receive power on 1/1/3/1 is -23.0 dBm, in the normal range.

```
zSH> gpononu power show 1/3
Processing list of 64
This command may take several minutes to complete.
Do you want to continue?  (yes or no) [no] yes
                 OLT             ONT
Interface    Receive Power   Receive Power
==========   =============   =============
1-1-3-1        -17.4 dBm       -23.0 dBm
1-1-3-2        -17.2 dBm       -23.0 dBm
1-1-3-3        -18.2 dBm       -23.0 dBm
```

```
1-1-3-4        -18.0 dBm       -23.0 dBm
1-1-3-5        -17.4 dBm       -23.0 dBm
1-1-3-6        -17.8 dBm       -24.0 dBm
1-1-3-7        -16.9 dBm       -23.0 dBm
1-1-3-8        -17.4 dBm       -23.0 dBm
1-1-3-9        -17.3 dBm       -23.0 dBm
1-1-3-10       -17.2 dBm       -22.0 dBm
1-1-3-11       error           -25.0 dBm
1-1-3-12       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.

2 The **gpononu status** command can display the same information. It displays upstream optical power level received at the OLT in the *OLT Rx Power* column and downstream optical power level received at the ONU in the *ONT Rx Power* column.

```
zSH> gpononu status 1/3/1
                            Omci         Gpon      Download      OLT      ONT        Distance
ID    Onu     OperStatus  ConfigState   OnuStatus     State     Rx Power  Rx Power      (KM)
=== ======== ============ ============ =========== ============ ========= ========= =======
2   1-1-1-2      Up         Active        Active     NoUpgrade  -17.4 dBm -23.0 dBm       0.0
```

Viewing the transmit parameters on OLT

Digital Diagnostic Monitoring (DDM) provides diagnostic information about the SFP. With DDM function, the SFP optical transceiver 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.

Table 25 provides the output fields description for this command.

Table 25: Gponolt port show Command Output Field Explanations

Field	Description
SLOT/OLT	The GPON ID.
Temperature	Internally measured Transceiver Temperature of the OLT in Celsius.
Voltage	Internally measured Transceiver Supply Voltage of the OLT in Volts.
Tx Bias Current	Measured Tx Bias current per OLT in Milli Amperes.

Table 25: Gponolt port show Command Output Field Explanations

Field	Description
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
======== ============ ======= =============== ======== ==================
 1/1         NA          NA           NA          NA    SFP not present
 1/2         NA          NA           NA          NA    SFP not present
 1/3        43c        3.3v         10mA       3.0dBm    Ok
 1/4         NA          NA           NA          NA    SFP not present
 1/5         NA          NA           NA          NA    SFP not present
 1/6         NA          NA           NA          NA    SFP not present
 1/7         NA          NA           NA          NA    SFP not present
 1/8         NA          NA           NA          NA    SFP not present
```

Manage ONU with OMCI

This section describes how GPON ONUs are managed with OMCI (standard G988):

Topics:

- View OMCI configuration states and alarms on ONU, page 453
- Retrieve status of subscriber facing ports, page 455
- Administration of subscriber facing ports, page 455
- Configurable speed of subscriber facing ports, page 456
- Configurable Transmit Gain, Receive Gain, and Fax Mode parameters for POTS ports (Smart OMCI Only), page 457
- Manual upgrade on an ONU, page 459
- Auto upgrade on an ONU, page 462
- View the ONU upgrade status, page 465
- Reboot an ONU, page 467

- Re-synchronize an ONU, page 468
- Add new services for a subscriber without affecting existing services (Smart OMCI Only), page 468
- Retrieve alarm information on an ONU, page 468
- View or change trap reporting status on an ONU, page 468
- Delete ONU configuration, page 469

View OMCI configuration states and alarms on ONU

View OMCI configuration states and alarms generated on an ONU with the **gpononu status** command.

Table 26 provides the output fields description for this command.

Table 26: 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**
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.

Table 26: Gpononu status Command Output Field Explanations

Field	Description
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
DownloadState	ONT software image download states. Values: — this ONU is not configured with OMCI **None** **Queued** **NoUpgrade** **Downloading** **Complete** **Error** **Aborted**
OLT Rx Power	Upstream optical power level received at the OLT.
ONT Rx Power	Downstream optical power level received at the ONU.
Distance (KM)	Calculated distance of an ONU from the OLT.

This example shows an ONU that is enabled and completes the OMCI provisioning.

```
zSH> gpononu status 1/1/1

                                Omci        Gpon     Download     OLT       ONT      Distance
ID     Onu    OperStatus ConfigState OnuStatus  State     Rx Power  Rx Power   (KM)
```

```
== ======== ========== =========== ========= ========= ========= ========= =======
1  1-1-1-1     Up     Done         Active   NoUpgrade -23.8 dBm -23.0 dBm   18
```

This example shows an ONU is enabled and then goes down with a dying gasp.

```
zSH> gpononu status 11/7/1

                        Omci      Gpon                   Download     OLT      ONT Distance
ID    Onu   OperStatus ConfigState OnuStatus               State     Rx Power Rx Power  (KM)
== ======== ========== =========== ====================== ========= ========= ======== ====
1  1-1-7-1    Down     Done      Inactive+LOS+LOF+DG+OAML NoUpgrade -23.8dBm -23.0dBm   18
```

Retrieve status of subscriber facing ports

From the MXK-194/198, 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 1/1/1 port eth 1
   1/1/1 ONU Port Status
           Ethernet Port Status - Port 1
                   Administrative State            up
                   Operational State               active
```

Administration of subscriber facing ports

Enabling or disabling admin state of a subscriber facing port

From the MXK-194/198, 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).

Note that for Dynamic OMCI, this can also be done in CPE subscriber profiles.

1. Set an ONU ethernet port admin state to down:

   ```
   zSH> gpononu port 1/1/1 eth 1 down
   ```

2. Verify the admin status:

```
zSH> gpononu status 1/1/1 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 1/1/1 eth 1 up
```

4 Verify the admin state is up:

```
zSH> gpononu status 1/1/1 port eth 1
1/1/1 ONU Port Status
        Ethernet Port Status - Port 1
                Administrative State            up
                Operational State               active
```

Configurable speed of subscriber facing 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

Note that for Dynamic OMCI, this can also be done in CPE subscriber profiles.

The following example configures an ONU Ethernet subscriber port for 1000 full duplex mode:

```
zSH> gpononu auto-detect 1/1/1 eth 1 1000F
```

Configurable Transmit Gain, Receive Gain, and Fax Mode parameters for POTS ports (Smart OMCI Only)

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 profile 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 profile update gen** command, then enter the corresponding variable indexes and values.

The following example shows how to configure these parameters.

```
zSH> gpononu profile show gen
2520-GEN
zSH> gpononu profile 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.zhone.com]”  metaswitch.oak.zhone.com
  52  “SipRegistrar [172.16.60.51]”  metaswitch.oak.zhone.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-194/198 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-194/198 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 by the 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, the ONU automatically checks whether the image file is valid. Only valid files can be activated.

Upgrade the firmware image on an ONU from the MXK-194/198 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#*]

Table 27 provides the description for command options in the **gpononu image** command.

Table 27: Gpononu image Command Option Explanations

Command Option	Description
download *filename* [**part** *partition#*]	Download an image file to the ONU from the OLT. Part partition number is optional. An image file will be downloaded to either an inactive partition or an uncommitted partition. After downloading, the ONU validates the file.
activate [**part** *partition#*]	Bootup a valid file in the inactive partition immediately in the ONU. Part partition number is optional. Only one partition at a time can be active.
commit [**part** *partition#*]	Specify a default file to bootup the 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 can be committed.

Table 27: Gpononu image Command Option Explanations

Command Option	Description
download-activate *filename* [**part** *partition#*]	Perform the download action, and then if the file passes the validation check, perform the activate action. Part partition number is optional.
download-activate-commit *filename* [**part** *partition#*]	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.
part *partition#*	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-194/198 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 exists in the MXK-194/198 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 1/4/24 download-activate-commit
   2510/2510.img
   Image "/card1/2510/2510.img" downloading to onu 1/4/
   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 1/4/24 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 1/4/24 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 1/4/24 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 1/4/24 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 1/4/24 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-194/198 users 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-194/198 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-194/198 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-194/198 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  1-1-1-1/gpononu
gpon-olt-onu-config  1-1-1-1/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}
us-rx-power-monitoring-mode: ---------->  {monitoronly}
us-rx-power-high-threshold: ----------->  {-10}
us-rx-power-low-threshold: ------------>  {-30}
dba-status-reporting: ----------------->  {disabled}
```

Or you can view it with the **gpononu auto-upgrade show** *slot* [/*olt*[/*onu*]] | **all** command:

```
zSH> gpononu auto-upgrade show 1
Processing list of 512
This command may take several minutes to complete.
Do you want to continue?  (yes or no) [no] yes
Slot 1 olt 1
ONU       Name        Auto-upgrade
=== ================= =============
  1          1-1-1-1     enabled
  2          1-1-1-2     enabled
  3          1-1-1-3     enabled
  4          1-1-1-4     enabled
  5          1-1-1-5     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 1/1
This command may affect many OLTs and ONUs.
```

```
Do you want to continue?  (yes or no) [no] yes

processing 1-1-1-1/gpononu ...
processing 1-1-1-2/gpononu ...
processing 1-1-1-3/gpononu ...
processing 1-1-1-4/gpononu ...
processing 1-1-1-5/gpononu ...
...
Operation successful.
```

3 Disable ONU auto-upgrade.

Disabling auto-upgrade in the auto-upgrade template or on the individual ONU causes the MXK-194/198 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 1/1
This command may affect many OLTs and ONUs.
Do you want to continue?  (yes or no) [no] yes

processing 1-1-1-1/gpononu ...
processing 1-1-1-2/gpononu ...
processing 1-1-1-3/gpononu ...
processing 1-1-1-4/gpononu ...
processing 1-1-1-5/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 of 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:

1 View the upgrade status on an ONU:

```
zSH> gpononu upgrade show 1/4/2
ONU      State          Model                Start Time       Act Cmt Typ Part
-------------------------------------------------------------------------------
1/4/2   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 probable 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

2 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
-------------------------------------------------------------------------------

1/4/1   Complete       2510              OCT 01 22:28:40 2009  F  F  Manl 1
1/4/2   Complete       2510              OCT 01 19:37:39 2009  F  F  Auto 1
```

3 View the upgrade status and the partition information on an ONU:

```
zSH> gpononu image 1/4/2 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 1/4
 ONU      State          Model                Start Time      Act Cmt Typ P
-------------------------------------------------------------------------
1/4/11  NoUpgrade   2510
1/4/12  NoUpgrade   2510
1/4/13  NoUpgrade   2510
1/4/14  NoUpgrade   2510
1/4/15  NoUpgrade   2510
1/4/16  NoUpgrade   2510
1/4/17  NoUpgrade   2510
1/4/18  NoUpgrade   2510
1/4/19  NoUpgrade   2510
1/4/20  NoUpgrade   2510
1/4/21  NoUpgrade   2510
1/4/22  NoUpgrade   2510
1/4/23  NoUpgrade   2510
1/4/25  NoUpgrade   2510
1/4/26  NoUpgrade   2510
1/4/27  NoUpgrade   2510
1/4/28  NoUpgrade   2510
1/4/29  NoUpgrade   2510
1/4/30  NoUpgrade   2510
1/4/31  NoUpgrade   2510
1/4/32  NoUpgrade   2510
1/4/1   Complete    2510                OCT 01 22:28:40 2009  F  F  Manl 1
1/4/2   Complete    2510                OCT 01 19:37:39 2009  F  F  Auto 1
1/4/3   Aborted     2510                OCT 01 19:41:37 2009  F  F  Auto 1
1/4/4   Aborted     2510                OCT 01 19:41:38 2009  F  F  Auto 1
1/4/5   Aborted     2510                OCT 01 19:41:42 2009  F  F  Auto 1
1/4/6   Aborted     2510                OCT 01 19:41:46 2009  F  F  Auto 1
1/4/7   Aborted     2510                OCT 01 19:37:59 2009  F  F  Auto 1
1/4/8   Aborted     2510                OCT 01 19:38:00 2009  F  F  Auto 1
1/4/9   Aborted     2510                OCT 01 19:38:05 2009  F  F  Auto 0
1/4/10  Aborted     2510                OCT 01 19:38:10 2009  F  F  Auto 0
1/4/24  Downloaded  2510                OCT 01 23:41:57 2009  F  F  Manl 1
```

Reboot an ONU

Reboot the remote ONU from the MXK-194/198 with the **gpononu reboot** command.

Reboot an ONU:

```
zSH> gpononu reboot 1/4/2
```

Re-synchronize an ONU

Synchronize an ONU with the MXK-194/198 with the gpononu resync command. This command causes the MXK-194/198 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 in Smart OMCI configuration. It is not common to use it in Dynamic OMCI. It is typically only used for debug.

Re-sync an ONU:

```
zSH> gpononu resync 1/4/2
```

Add new services for a subscriber without affecting existing services (Smart OMCI Only)

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.

Note: If the ONU is provisioned by Dynamic OMCI, you don't need to use the **gpononu apply** command. The new services will be updated to the ONU automatically.

Users can modify the Specific profile or Generic profile and issue the following command:

```
zSH> gpononu apply 1/4/2
```

The **gpononu apply** command issues the OMCI configuration command in the ME profile. This command does not force a resync of the ONU. If a 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 1/4/2
   1/4/2 ONU Active Alarms
           PptpEthUni 0x0402 LanLos
           PptpEthUni 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-194/198 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-194/198, 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 1/4/2
   Slot 1 olt 4
   ONU       Name        PhysicalTraps   OntTraps    LineStatusTraps
   === ================= =============   =========   ===============
     2          1-1-4-2     enabled      disabled          auto
```

Change the current reporting status of traps on ONU 1/4/2 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 1/4/2 phy
```

Delete ONU configuration

To remove all the ONU configurations on an ONU, and set this ONU to defaults, you can use the **gpononu delete *slot*[/*olt*[/*onu*]]**command.

Note that the **gpononu delete** command will

- delete all CPE subscriber profiles and CPE connections that were created on the ONU,
- delete the ONU's OMCI Specific profile (for Smart OMCI, if it exists),
- delete the MXK-194/198 bridges that were created on the GEM port, and GEM ports that were created on that ONU,
- set the onu related fields in the gpon-olt-config profile, and the gpon-olt-onu-config profile of the ONU to defaults,
- set the adminstatus, ifName, and redundancy-param1 fields in the ONU I/F translate profile to defaults.

If you want to delete services on ONUs with Dynamic OMCI, use the **cpe delete** *slot*[/*olt*[/*onu*]] command. This command deletes all CPE subscriber profiles on the ONU. For the details, refer to *Deleting CPE subscriber profiles on an ONU* on page 412.

Deleting ONU configuration on an ONU

This example deletes the ONU configuration on ONU 1/3/1, and verifies it:

1 Remove the ONU configuration on ONU 1/3/1.

```
zSH> gpononu delete 1/3/1

Ok to delete ONU 1/3/1 and all of it's configuration? [yes] or [no]: yes

Do you want to exit from this request? [yes] or [no]: no

Are you sure? [yes] or [no]: yes

deleting ONU 1/3/1

ONU 1/3/1 has been deleted
```

2 Verify that the MXK-194/198 bridges that were created on the GEM ports of ONU 1/3/1 are removed.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                  Bridge                        St  Table Data
-----------------------------------------------------------------------------------------------------------
dwn             Tagged  999    1/1/1/2/gpononu           1-3-1-257-gponport-999/bridge  UP
upl             ST    2/998    1/1/4/0/eth               ethernet4-2-998/bridge         UP  S SLAN 998 VLAN 2 default
upl             ST    3/998    1/1/4/0/eth               ethernet4-3-998/bridge         UP  S SLAN 998 VLAN 3 default
tls             Tagged  300    1/1/4/0/eth               ethernet4-300/bridge           UP
tls             Tagged  503    1/1/4/0/eth               ethernet4-503/bridge           UP
upl             Tagged  999    1/1/4/0/eth               ethernet4-999/bridge           UP  S VLAN 999 default
upl                1001        1/1/8/0/eth               ethernet8/bridge               UP  S VLAN 1001 default
7 Bridge Interfaces displayed
```

3 Verify that the GEM ports that were created on ONU 1/3/1 are removed.

```
zSH> gpononu gemports 1/3/1
                                                Fixed UBR Fixed CBR Assured   Max       Extra
                                traf            Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU           GEM Port   Admin prof   compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type       allocId DBA
========== ============ ===== ====== ===== ===== ========= ========= ========= ========= ========== ======= =====
```

4 Verify that the CPE connections that were created on the Uni-ports of ONU 1/3/1 are removed.

```
zSH> bridge show onu
           GEM   ONU     ONU UNI    OLT           OLT
ONU        Port  UNI     VLAN/SLAN  VLAN/SLAN     MVR   Service  OLT Bridge                                ST
-----------------------------------------------------------------------------------------------------------------
1/3/1        257   eth 1              Tagged  999         iptv     1-3-1-257-gponport-999/bridge           UP
1 Bridge Interfaces displayed
1 GPON ONU Connections displayed
```

5 Verify the CPE subscriber profiles are removed on this ONU.

```
zSH> CPE> ETH> show 1/3/1
No services found.

zSH> CPE> VOIP> show 1/3/1
No services found.

zSH> CPE> PWE> show 1/3/1
No services found.
```

6 To verify that onu related fields in the gpon-olt-config profile, and the gpon-olt-onu-config profile are set to defaults, use the following commands.

```
zSH> get gpon-olt-config 1-1-3-0/gponolt

zSH> get gpon-olt-onu-config 1-1-3-1/gpononu
```

7 Verify that adminstatus, ifName, and redundancy-param1 fields in the if-translation profile of ONU 1/3/1 are set to defaults.

```
zSH> get if-translate 1-1-3-1/gpononu
if-translate  1-1-3-1/gpononu
ifIndex: ----------->  {175}
shelf: ------------->  {1}
slot: -------------->  {1}
port: -------------->  {3}
subport: ----------->  {1}
type: -------------->  {other}
adminstatus: ------->  {up}
physical-flag: ----->  {false}
iftype-extension: -->  {gpononu}
ifName: ------------>  {1-1-3-1}
redundancy-param1: ->  {0}
description-index: ->  {0}
```

Bandwidth allocation on T-cont

The bandwidth allocation for upstream traffic from the ONU to the MXK-194/198 is configured in the GPON traffic profile.

This section includes the following topics:

- Configure GPON traffic profile, page 472
- Dynamic Bandwidth Allocation (DBA), page 481

Configure GPON traffic profile

The general function of a GPON traffic profile is to set a maximum upstream transmission rate, class of service types, and bandwidth allocation type per T-cont on the PON. When creating a GEM port with the bridge add command, you must associate a GPON traffic profile with it. To minimize the amount of configuration per subscriber, the GPON traffic profile is defined in the system, and has an index. The index of a GPON traffic profile is used in the creation of a subscriber bridge interface. Thus, if many customers want the same service, one GPON traffic profile can be created, and its index could be used for multiple GEM ports without having to define the bandwidth parameters every time a subscriber is provisioned.

For the detail DBA configuration on the GPON traffic profile, refer to section *Dynamic Bandwidth Allocation (DBA)* on page 481.

The CAC validation is performed during the GPON traffic profile configuration.

Creating a GPON traffic profile

Table 28 provides descriptions for GPON traffic profile parameters.

Table 28: GPON Traffic Profile parameters description

Parameter	Description
guaranteed-upstream-bw	Specifies the guaranteed non-DBA upstream bandwidth on a T-cont. The value should be a multiple of 512 and cannot exceed 1,048,576 Kbps. Note that the guaranteed-upstream-bw is for CBR and UBR only and is not used in DBA. If DBA is disabled the guaranteed upstream bandwidth should be non-zero.
traffic-class	Specifies the upstream traffic class type of a T-cont. 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. **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.
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 not used for DBA. Default: **false** Values: **true** **false**
shared	Shared feature is to let the GEM ports under the same ONU share the upstream bandwidth. Select true if the GEM port which uses this traffic descriptor shares a T-CONT (i.e. 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**

Table 28: GPON Traffic Profile parameters description

Parameter	Description
dba-enable	Enable or disable DBA. The default value is false. Default: **false** Values: **true** **false**
dba-fixed-us-ubr-bw	Guaranteed bandwidth in upstream direction for UBR type of traffic. It is applicable only 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	Guaranteed bandwidth in upstream direction for CBR type of traffic. It is applicable only 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 in upstream direction. It is applicable only when DBA is enabled. It will be allocated when traffic demand exists, it may not be given without demand. 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 28: GPON Traffic Profile parameters description

Parameter	Description
dba-max-us-bw	This is the maximum DBA bandwidth that can be allocated to a T-CONT. This maximum bandwidth includes guaranteed bandwidth and non-guaranteed bandwidth. You can 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** Bandwidth only given if bandwidth is available but not guaranteed. Nonassured has higher priority for getting unused bandwidth than besteffort. **besteffort** Demand only met if remaining upstream bandwidth is available. Besteffort has the lowest priority.

1 Create a GPON traffic profile with the **new gpon-traffic-profile** *index* command.

The 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 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 GPON traffic 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: ->  {0}: 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 GPON traffic profile profile is to let the GEM ports that under same ONU 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 1-1-2-501 and 1-1-2-701 have different Alloc-Ids, 501 and 701.

```
zSH> gpononu gemports 1/2

Processing list of 64

This command may take several minutes to complete.
         Do you want to continue?  (yes or no) [no] yes
                                   Fixed UBR Fixed CBR Assured   Max       Extra
                     traf           Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU     GEM Port Admin prof compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type
allocId DBA
======= ========= ===== ==== ===== ===== ========= ========= ========= ========= ==========
======= =====
1-1-2-1 1-1-2-501 Up 1024 False False 1.024      0       n/a      n/a      n/a
501    n/a
        1-1-2-701 Up 1024 False False 1.024      0       n/a      n/a      n/a
701    n/a
1-1-2-2 1-1-2-502 Up 1024 False False 1.024      0       n/a      n/a      n/a
502    n/a
        1-1-2-702 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 GPON traffic profile 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: ---------->  {ubr}:
compensated: ------------>  {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 GPON traffic profile to multiple GEM ports by using the **bridge add** or **host add** command.

4 List all the ONU GEM ports use this GPON traffic profile.

```
zSH> gpononu gtp list 512

To Abort the operation enter Ctrl-C

GEM Ports that use Traffic Profile 512

ONU Interface    GEM Port
=============  ==============
1-1-1-1        1-1-1-501
1-1-1-1        1-1-1-701
1-1-1-2        1-1-1-502
1-1-1-2        1-1-1-702
```

5 View the Alloc-Id values assigned to the GEM ports when shared feature is enabled.

This example shows GEM ports 1-1-1-501 and 1-1-1-701 have the same Alloc-Ids, 501.

```
zSH> gpononu gemports 1/1

Processing list of 64

This command may take several minutes to complete.
         Do you want to continue?  (yes or no) [no] yes
                                   Fixed UBR Fixed CBR Assured   Max       Extra
                      traf          Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU     GEM Port Admin prof compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type
allocId DBA
======= ========= ===== ==== ===== ===== ========= ========= ========= ========= ==========
======= =====
1-1-1-1 1-1-1-501 Up 512 False True     0.5        0     n/a      n/a      n/a
501    n/a
        1-1-1-701 Up 512 False True     0.5        0     n/a      n/a      n/a
501    n/a
1-1-1-2 1-1-1-502 Up 512 False True     0.5        0     n/a      n/a      n/a
502    n/a
        1-1-1-702 Up 512 False True     0.5        0     n/a      n/a      n/a
502    n/a
```

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 an ONU GEM port. A GPON traffic profile is considered as “in-use” if it is already assigned to a GEM port. If a GPON traffic profile is in-use, the GPON traffic profile 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 GPON traffic profile is in use, the deletion will be rejected by the profile validation, and an error message will appear.

```
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
=============  ==============
1-1-1-1        1-1-1-501
1-1-7-3        1-1-7-503
1-1-7-4        1-1-7-504
1-1-7-5        1-1-7-505
```

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 1-1-1-501/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.

Enabling and Configuring DBA bandwidth on T-conts

User can enable or disable DBA and configure the DBA bandwidth on each T-cont with 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 28 on page 473.

1. Create a GPON-traffic-profile with DBA-related parameters. DBA is enabled in this GTP:

   ```
   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}:Not used in 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
New record saved.
```

2 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}:
```

3 Apply the DBA enabled GTP to an GEM port.

```
zSH> bridge add 1-1-1-5/gpononu gem 501 gtp 430080 downlink vlan 100 tagged
Adding bridge on 1-1-1-5/gpononu
Created bridge-interface-record 1-1-1-501-gponport-100/bridge
```

4 View the DBA bandwidth settings on GEMs on an ONU.

The allocID and DBA type will not be displayed until the ONU is activated.

```
zSH> gpononu gemports 1/1/5
                                                Fixed UBR Fixed CBR Assured   Max       Extra
                                 traf           Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU           GEM Port   Admin prof   compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type       allocId DBA
=========== ============ ===== ====== ===== ===== ========= ========= ========= ========= ========== ======= =====
              1-1-1-501    Up 430080 False False     0.512         0     1.024      1024 Nonassured       56 SR
```

Changing the DBA type per ONU

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.

By default, DBA type on each ONU is NSR. Users can change the DBA type to SR only if the ONU is inactive.

Note: Before changing the DBA type of an ONU from the default type NSR to SR, make sure the ONU supports SR.

Note: The only way to change the DBA type on an activated ONU is to clear and re-activate the ONU for the change to take effect.

This example changes DBA type of an activated ONU from NSR to SR.

1 View the DBA type on GEMs on an activated ONU.

```
zSH> gpononu gemports 1/1/5
                                                Fixed UBR Fixed CBR Assured   Max       Extra
                                traf            Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU           GEM Port   Admin prof   compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type       allocId DBA
=========== ============ ===== ====== ===== ===== ========= ========= ========= ========= ========== ======= =====
               1-1-1-501    Up 430080 False False     0.512         0     1.024      1024 Nonassured       56 SR
```

2 Enable the DBA type to SR on the ONU.

```
zSH> update gpon-olt-onu-config 1-1-1-5/gpononu
gpon-olt-onu-config  1-1-1-5/gpononu
Please provide the following: [q]uit.
serial-no-vendor-id: ------------------>  {ZNTS}:  **
read-only **
serial-no-vendor-specific: ------------>  {0}:  ** read-only
**
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}:  ** read-only
**
use-reg-id: ---------------------------->  {disabled}:
us-rx-power-monitoring-mode: ---------->  {monitoronly}:
us-rx-power-high-threshold: ----------->  {-10}:
us-rx-power-low-threshold: ------------>  {-30}:
dba-status-reporting: ----------------->  {disabled}:enabled
....................
```

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

3 De-register this ONU.

```
zSH> gpononu clear 1/1/5
```

4 Display the ONUs currently on the OLT, and discover the available serial numbers.

```
zSH> gpononu show 1/1
Free ONUs for slot 3 olt 1:
    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 1 olt 1:
sernoID   Vendor  Serial Number     sernoID   Vendor  Serial Number
    1      ZNTS    138543368
```

5 Activate this ONU.

```
zSH> gpononu set 1/1/5 1
Onu 5 successfully enabled with serial number ZNTS
138543368
```

GEM port creation

This section includes the following topics:

- Create a GEM port, page 484
- View the GEM port-related information, page 490

Create a GEM port

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-194/198 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-194/198 and the upstream data/voice/video source.

Before creating a GEM port, users must create a GPON Traffic Profile. The GTP provides the rate limiting on the T-cont where the GEM port is connected. For details on creating a GTP, refer to *Configure GPON traffic profile* on page 472. 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.

The ONU in this example 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 details, refer to *Chapter 8, Bridging Configuration,* on page 203.

For more information on how to configure video bridging, see *MXK bridged video* on page 373.

1 Create a bridging configuration for data services:

zSH> **bridge add 1-1-2-0/eth uplink vlan 100** *uplink bridge*

zSH> **bridge add 1-1-1-501/gponport gtp 1 downlink vlan 100 tagged** *downlink GEM port*

2 Create a bridging configuration for voice services:

zSH> **bridge add 1-1-2-0/eth uplink vlan 200** *uplink bridge*

zSH> **bridge add 1-1-1-701/gponport gtp 2 downlink vlan 200 tagged** *downlink GEM port*

3 Create a bridging configuration for video services:

zSH> **bridge add 1-1-2-0/eth uplink vlan 300** *uplink bridge*

zSH> **bridge-path modify ethernet2-300/bridge vlan 300 default igmptimer 30 igmpsnooping enable** *uplink bridge path*

zSH> **bridge add 1-13-1-901/gponport gtp 3 downlink vlan 300 tagged video 0/6** *downlink GEM port*

By specifying video 0/6, the downlink bridge will pass all IP multicast.

4 View the newly created GEM ports and associated traffic profiles for selected ONU with the **gpononu gemports** command.

```
zSH> gpononu gemports 1/1/1
                                    Fixed UBR Fixed CBR Assured   Max       Extra
                     traf            Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU     GEM Port Admin prof compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type
allocId DBA
======= ============== ===== ====== ===== ========= ========= ========= ========= =========
========== ======= =====
```

```
1-1-1-1 1-1-1-501 Up  1 False False   0.512       0      n/a     n/a      n/a
501    n/a
        1-1-1-701 Up  2 True False        0   0.512      n/a     n/a      n/a
701    n/a
        1-1-1-901 Up  3 True False        0   0.512      n/a     n/a      n/a
901    n/a
```

Creating host-based routes on GEM ports for data and voice 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-194/198 and downlink ONUs. This section also describes how to create a route on the uplink card to pass traffic between the MXK-194/198 and the upstream data/voice source.

The following examples create host-based routes on GEM ports, and configure the MXK-194/198 as a local DHCP server.

For completed network-based and host-based routings on GPON, refer to *Chapter 7, IP Configuration,* on page 131.

1 Create routed data services

zSH> **interface add 1-1-2-0/eth vlan 100 125.1.1.1/24** *creates IP interface on uplink*

zSH> **route add default 125.1.1.254 1** *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-194/198, data source, and DHCP server*

```
zSH> new dhcp-server-subnet 1  runs DHCP locally on MXK-194/198
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 1-1-1-501/gponport gtp 1 vlan 500 dynamic 1 1** *Creates downlink GEM port to pass data traffic between MXK-194/198 and ONU*

The above example creates GEM 501 on ONU 1/1/1/1 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 1-1-2-0/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 1-1-1-701/gponport gtp 2 vlan 700 dynamic 2 1***Creates downlink GEM port to pass voice traffic between MXK-194/198 and ONU*

The above example creates GEM 701 on ONU 1/1/1/1 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 View the newly created GEM ports and associated traffic profiles for the selected ONU with the **gpononu gemports** command.

```
zSH> gpononu gemports 1/1/1
                                    Fixed UBR Fixed CBR Assured   Max       Extra
                      traf           Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU     GEM Port Admin prof compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type
allocId DBA
======= ============== ===== ====== ===== ========= ========= ========= ========= =========
========== ======= =====
1-1-1-1 1-1-1-501 Up  1 False False   0.512       0      n/a     n/a      n/a
501    n/a
        1-1-1-701 Up  2 True False        0   0.512      n/a     n/a      n/a
701    n/a
```

Connection Admission Control (CAC) validation during the bridge/host creation on a GEM port

When using the **bridge add, interface 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 the user tried to 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 the 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, interface add** and **host add** command to create a GEM port, the user can use the following two commands to check the available Alloc-Ids and available bandwidth on the GPON physical port.

1. Check the available Alloc-Ids on a GPON physical port with the **gponolt status gtp** command:

```
zSH> gponolt status gtp
                                        DBA          Total
                                        Alloc-Ids    Alloc-Ids
OLT Interface  OLT State  # GEM Ports  used  avail  used  avail
=============  =========  ===========  ===========  ===========
1/1            Active         128        0    384   128    640
1/2            Active          63        0    384    63    705
1/3            Ready            2        0    384     2    766
1/4            Ready            0        0    384     0    768
1/5            Ready            0        0    384     0    768
1/6            Inactive         0        0    384     0    768
1/7            Inactive         0        0    384     0    768
1/8            Inactive         0        0    384     0    768
```

   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.

2. Check the available bandwidth on a GPON physical port with the **gponolt show bw** command:

```
zSH> gponolt show bw 1/1
SLOT 1/OLT 1:
        Total Available BW....................... 1029120 Kbps
        Total Available BW for Compensated CBR... 464076 Kbps
        Allocated UBR BW......................... 131072 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 gemports** command.

```
zSH> gpononu gemports 1/1/1
                                                   Fixed UBR Fixed CBR Assured   Max       Extra
                                                   Bandwidth Bandwidth Bandwidth Bandwidth Bandwidth
ONU           GEM Port   Admin traf prof   compn share Mbits/sec Mbits/sec Mbits/sec Mbits/sec Type       allocId DBA
=========== ============ ===== ====== ===== ===== ========= ========= ========= ========= ========== ======= ====        =
    1-1-1-1    1-1-1-501    Up      1 False False     1.024         0       n/a       n/a        n/a     501 n/a
               1-1-1-701    Up      1 False False     1.024         0       n/a       n/a        n/a     640 n/a
```

Table 29: Gpononu gemports 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.
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.

Table 29: Gpononu gemports Command Output Field Explanations

Field	Description
DBA	The DBA type. 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. NSR is the default value. **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

OMCI Statistics

The MXK-194/198 obtains ONU statistics from the ONU using OMCI. The MXK-194/198 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. These statistics are 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

The system retrieves the statistics collected during the previous 15 minutes interval. Without **previous**, the 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 1/4/2
1/4/2 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 30 defines the OMCI statistics displayed in the **gpononu statistics** command.

Table 30: OMCI statistics attributes

Attribute	Description
Interval end time	This attribute identifies the most recently finished 15-minute interval.
Threshold data pointer	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.
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.

GPON Alarms

GPON BIP Threshold Crossing Monitor Alarms

Users can monitor BIP threshold crossing alarms, set the threshold for BIP errors on GPON, and also configure whether or not to auto-disable the ONU if the threshold has been exceeded. BIP is a counter representing bit errors on the PON link to a specific ONU. This is configured on a per-OLT basis, but is monitored per ONU. To configure the GPON BIP threshold on all ONUs under an OLT, use the **update gpon-olt-config** command.

```
MXK-13> update gpon-olt-config 1-1-1-0/gponolt
 gpon-olt-config  1-1-1-0/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}:
sr-dba-reporting-block-size: ->  {48}:
protection-switchover-timer: ->  {500}:
preamble-override: ----------->  {disabled}:
preamble-type-0: ------------->  {0x00}:
preamble-type-1: ------------->  {0x00}:
preamble-type-3-pre-range: --->  {0x0b}:
preamble-type-3-post-range: -->  {0x08}:
preamble-type-3-pattern: ----->  {0xaa}:
bip-error-monitoring-mode: --->  {monitorOnly}:
```

```
errors-per-sample-threshold: ->  {100}:
errored-samples-threshold: --->  {10}:
bip-max-sample-gap: ---------->  {10}:
....................
Save changes? [s]ave, [c]hange or [q]uit:q
```

Table 31: BIP error threshold attributes in gpon-olt-config profile

Attribute	Description
bip-error-monitoring-mode	Disable or enable the BIP error monitoring. Values: **disabled** The BIP error monitoring feature is disabled. **monitorOnly** Monitor BIP errors. When the ONU crosses the BIP error threshold, trigger a local alarm and send a trap to ZMS. **blockOnError** Monitor BIP errors. When the ONU crosses the BIP error threshold, trigger a local alarm , send a trap to ZMS, disable ONU ranging and set ONU line status to DSA (i.e. disabled). Default: **monitorOnly**
errors-per-sample-threshold	If the number of BIP errors per sample exceeds this threshold, it is counted as an errored sample. Default: **100**
errored-samples-threshold	If the number of errored samples exceed this sample threshold, report and disable the onu if in blockOnError mode, otherwise simply report the threshold as being exceeded. Default: **10**
bip-max-sample-gap	If two adjacent errored samples were taken farther apart than this threshold, do not count the earlier sample as an errored sample. This value is in the unit of seconds. Default: **10**

Configuring GPON BIP error threshold crossing monitor alarms

1 View the ONU status.

```
MXK-13> onu status 1/1/2
                          Omci       Gpon    Download    OLT      ONT     Distance
ID    Onu   OperStatus ConfigState OnuStatus  State    Rx Power  Rx Power   (KM)
== ======== ========== =========== ========= ========= ========= ========= =======
2  1-1-1-2     Up     Active      Active    NoUpgrade -23.8 dBm -23.0 dBm   18
```

2 Configure the BIP error monitoring mode and thresholds as desired. This example changes the monitoring mode to **blockonerror**, and changes the BIP error threshold values.

```
MXK-13> update gpon-olt-config 1-1-1-0/gponolt
 gpon-olt-config  1-1-1-0/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}:
sr-dba-reporting-block-size: ->  {48}:
protection-switchover-timer: ->  {500}:
preamble-override: ----------->  {disabled}:
preamble-type-0: ------------->  {0x00}:
preamble-type-1: ------------->  {0x00}:
preamble-type-3-pre-range: --->  {0x0b}:
preamble-type-3-post-range: -->  {0x08}:
preamble-type-3-pattern: ----->  {0xaa}:
bip-error-monitoring-mode: --->  {monitorOnly}:blockonerror
errors-per-sample-threshold: ->  {100}: 99
errored-samples-threshold: --->  {10}:9
bip-max-sample-gap: ---------->  {10}:9
....................
Save changes? [s]ave, [c]hange or [q]uit:s
```

3 View the ONU status. This example assumes the BIP error on this ONU exceeded the threshold values. With the **blocknoerror** mode, the ONU will raise an alarm and be auto-disabled. The **GponOnuStatus** in this example shows a brief description about this ONU is inactive and EXCBIPDSA (i.e. exceeded BIP threshold, and ONU is disabled.).

```
MXK-13> onu status 1/1/2
                          Omci       Gpon             Download    OLT      ONT     Distance
ID    Onu   OperStatus ConfigState OnuStatus           State    Rx Power  Rx Power   (KM)
== ======== ========== =========== ================== ============ ====== ========= =======
2  1-1-1-2   Down     Inactive    Inactive+EXCBIPDSA       None   error      error   0.0
```

GponOnuStatus acronym definitions:

"Active" - ONU is Active

"Inactive" - ONU is Inactive

"LOS" - Loss of Signal

"LOF" - Loss of Frame

"DOW" - Drift Of Window

"SF" - Signal Fail

"SD" - Signal Degrade

"LCDG" - Loss of GEM channel delineation

"RD" - Remote defect

"TF" - Trasmitter Failure

"SUF" - Start-up Failure

"LOA" - Loss of Acknowledge

"DG" - Receive Dying-gasp

"OAML" - PLOAM Cell Loss

"MEM" - Message Error Message

"PEE" - Physical Equipment Error

"EXCBIPDSA" - Disable Onu, excessive BIP errors

"EXCBIP" - Excessive BIP errors. ONU is not disabled

"RXPWRDSA" - Upstream Rx Power out of range. ONU is disabled.

"RXPWRNOTDSA" - Upstream Rx Power out of range. ONU is not disabled.

4 View the raised alarms on this ONU at the system level.

```
MXK-13> alarm show
************    Central Alarm Manager    ************
        ActiveAlarmCurrentCount         :8
        AlarmTotalCount                 :15
        ClearAlarmTotalCount            :7
        OverflowAlarmTableCount         :0
ResourceId              AlarmType                                AlarmSeverity
----------              ---------                                -------------
1-1-3-0/eth             linkDown                                    critical
1-1-1-0/gponolt         linkDown                                   critical
1-1-2-0/gponolt         linkDown                                   critical
1-1-3-0/gponolt         linkDown                                   critical
1-1-4-0/gponolt         linkDown                                   critical
1-1-1-2/gpononu         linkDown                                   minor
system                   not_in_redundant_mode                      major
1-1-1-2/gpononu         inactive,bip threshold exceeded,dsa        minor
```

Note: If more than one error condition is present (example: Excessive BIP errors and Optical Rx Power too high), the local alarm text may be too long to fit within the display output. In this case, AlarmType shows "issue 'onu status' command 0x300002" in order to prompt user to enter the **onu status** command for more details. The "0x300002" value is the actual alarm status word and will vary.

```
MXK-13> alarm show
************    Central Alarm Manager    ************
        ActiveAlarmCurrentCount         :8
        AlarmTotalCount                 :32
        ClearAlarmTotalCount            :24
        OverflowAlarmTableCount         :0
ResourceId               AlarmType                                  AlarmSeverity
----------               ---------                                  -------------
1-1-1-2/gpononu         issue 'onu status' command 0x300002         minor
```

GPON High and Low Receive Power Threshold Alarms

By default, the MXK-194/198 will trigger a local alarm, and send a trap to ZMS when the GPON high/low receive power thresholds are crossed for the ONU received power on the upstream. The default value of the High threshold is -10 dbm. The default value of the Low threshold is -30 dbm. Users can change the default threshold values, and choose the upstream received power monitoring mode as desired.

The GPON high/low receive power threshold values and monitoring modes are configured on a per-ONU basis with the **update gpon-olt-onu-config** command.

```
MXK-13> update gpon-olt-onu-config 1-1-1-2/gpononu
gpon-olt-onu-config  1-1-1-2/gpononu
Please provide the following: [q]uit.
serial-no-vendor-id: ------------------>  {ZNTS}:  ** read-only **
serial-no-vendor-specific: ------------>  {2216690777}:  ** read-only **
password: ----------------------------->  {}:
auto-learn: --------------------------->  {enabled}:
power-level: -------------------------->  {0}:
us-ber-interval: ---------------------->  {5000}:
ds-ber-interval: ---------------------->  {5000}:
onu-added: ---------------------------->  {true}:
omci-file-name: ----------------------->  {}:
ONU-Managed-Entity-Profile-name: ------>  {znid-gpon-2510-omci-4port-me}:
ONU-Generic-Assignments-Profile-name: ->  {znid-gpon-2510-omci-4port-gen}:
physical-traps: ----------------------->  {disabled}:
ont-traps: ---------------------------->  {disabled}:
line-status-traps: -------------------->  {disabled}:
auto-upgrade: ------------------------->  {enabled}:
serial-no-vendor-specific-fsan: ------->  {84200459}:  ** read-only **
use-reg-id: --------------------------->  {disabled}:
us-rx-power-monitoring-mode: ---------->  {monitorOnly}:
us-rx-power-high-threshold: ----------->  {-10}:
us-rx-power-low-threshold: ------------>  {-30}:
dba-status-reporting: ----------------->  {disabled}
.................
Save changes? [s]ave, [c]hange or [q]uit:q
```

Table 32: Received power threshold attributes in gpon-olt-onu-config profile

Attribute	Description
us-rx-power-monitoring-mode	Disable or enable the received power threshold alarm. Values: **disabled** This feature is disabled. **monitorOnly** Monitor ONU Receive Power Level. When ONU Receive Power Level crosses either the High or Low thresholds, trigger a local alarm, and send trap to ZMS. **blockOnError** Monitor ONU Receive Power Level. When ONU Receive Power Level crosses either the High or Low thresholds, trigger a local alarm, send trap to ZMS, disable ONT ranging and set ONT line status to DSA. Default: **monitorOnly**
us-rx-power-high-threshold	Upstream Receive Power High Threshold value, in the unit of dbm. Default: **-10**
us-rx-power-low-threshold	Upstream Receive Power Low Threshold value, in the unit of dbm. Default: **-30**

Configuring GPON high and low received power threshold alarms

1. View the ONU status. In this example, the upstream ONU received power under the OLT Rx Power column is -23.8 dBm, which is within the default value range of the GPON high and low received power threshold (-10 to -30).

```
MXK-13> onu status 1/1/2
                              Omci       Gpon    Download    OLT      ONT     Distance
ID    Onu    OperStatus ConfigState OnuStatus  State    Rx Power  Rx Power   (KM)
== ======== ========== =========== ========= ========= ========= ========= =======
2  1-1-1-2     Up     Active      Active    NoUpgrade -23.8 dBm -23.0 dBm   18
```

2. Configure the upstream ONU received power monitoring mode and thresholds as desired.

   This example changes the low-threshold to -20 from the default value -30, and changes the monitoring mode to **blockonerror.** If the current OLT RX power has crossed the low threshold, a received power threshold alarm will be triggered, and the ONU will be disabled.

```
MXK-13> update gpon-olt-onu-config 1-1-1-2/gpononu
gpon-olt-onu-config  1-1-1-2/gpononu
Please provide the following: [q]uit.
serial-no-vendor-id: ------------------>  {ZNTS}:  ** read-only **
serial-no-vendor-specific: ------------>  {2216690777}:  ** read-only **
password: ----------------------------->  {}:
auto-learn: --------------------------->  {enabled}:
power-level: -------------------------->  {0}:
us-ber-interval: ---------------------->  {5000}:
ds-ber-interval: ---------------------->  {5000}:
```

```
onu-added: ----------------------------->  {true}:
omci-file-name: ------------------------>  {}:
ONU-Managed-Entity-Profile-name: ------>  {znid-gpon-2510-omci-4port-me}:
ONU-Generic-Assignments-Profile-name: ->  {znid-gpon-2510-omci-4port-gen}:
physical-traps: ------------------------>  {disabled}:
ont-traps: ----------------------------->  {disabled}:
line-status-traps: --------------------->  {disabled}:
auto-upgrade: --------------------------->  {enabled}:
serial-no-vendor-specific-fsan: ------->  {84200459}:  ** read-only **
use-reg-id: ----------------------------->  {disabled}:
us-rx-power-monitoring-mode: ---------->  {monitorOnly}:blockonerror
us-rx-power-high-threshold: ----------->  {-10}:
us-rx-power-low-threshold: ------------>  {-30}:-20
dba-status-reporting: ----------------->  {disabled}
....................
Save changes? [s]ave, [c]hange or [q]uit:s
```

3 View the ONU status.

This example shows GponOnuStatus is inactive and RXPWRDSA (i.e. received power out of range, and ONU is disabled.) Refer to the **alarm show** command for the explanation of the cryptic acronyms.

```
MXK-13> onu status 1/1/2
                        Omci       Gpon            Download    OLT      ONT      Distance
ID    Onu   OperStatus ConfigState OnuStatus          State     Rx Power Rx Power   (KM)
== ======== ========== =========== ================= ========= ========= ========= =======
2  1-1-1-2     Down     Inactive  Inactive+RXPWRDSA   None       error    error     0.0
```

4 View the alarms on this ONU at the system level.

Two alarms are raised, link down and Rx power threshold alarms.

```
MXK-13> alarm show
************    Central Alarm Manager    ************
        ActiveAlarmCurrentCount         :8
        AlarmTotalCount                 :32
        ClearAlarmTotalCount            :24
        OverflowAlarmTableCount         :0
ResourceId               AlarmType                                  AlarmSeverity
----------               ---------                                  -------------
1-1-3-0/eth              linkDown                                      critical
1-1-1-0/gponolt         linkDown                                     critical
1-1-2-0/gponolt         linkDown                                     critical
1-1-3-0/gponolt         linkDown                                     critical
1-1-4-0/gponolt         linkDown                                     critical
1-1-1-2/gpononu         linkDown                                     minor
system                   not_in_redundant_mode                        major
1-1-1-2/gpononu         inactive,rx power out of range,dsa           minor
```

Rogue ONU Alarm and RSSI Rogue ONU Alarm

A rogue ONU is an ONU that transmits outside of its allocated bandwidth map. It may cause some severe problems such as its data cannot be received by the OLT as valid data, interfere with data from other ONUs, or bring down the OLT after it connected to that OLT.

To detect a rogue ONU, you can use the following two detections per OLT basis:

1. Background process detection

   To detect a rogue ONU, run the periodical background process detection first.

   The periodical background process opens a special allocation window and monitors for potential rogue transmission. The special window is opened with an unused Alloc_ID, for which no response is expected unless there is a rogue ONU. The window is opened so that it may detect a transmission either within the PON distance or further. When a rogue ONU transmission is detected in the special window, a **Rogue ONU alarm** and trap will be sent.

2. Rogue RSSI measurement

   The rogue RSSI measurement is a powerful tool for identifying the rogue ONU by measuring the ONU's transmission power. It is recommended to run the rogue RSSI detection after the periodical background process.

   The RSSI measurement is a stand-alone utility for testing rogue transmission when no upstream burst is expected. The intention is to identify when a rogue ONU injects a constant energy on the link, and does not respond to OLT allocations. If the rogue ONU RSSI measurement is higher than the rogue-onu-rx-power-threshold defined in the **gpon-olt-config** profile, a **RSSI Rogue ONU alarm** and trap will be sent.

   Caution: The rogue RSSI measurement is semi-invasive mode. During the activation of the RSSI measurement on an OLT port it is not allowed to provision Alloc_IDs or to activate ONUs under that OLT port.

   Note: Do not run the periodical background process and the rogue RSSI detection at the same time.

Both the rogue ONU alarm and the RSSI rogue ONU alarm have the severity level minor.

Users can configure the RSSI rogue measurement or background process detection in the gpon-olt-config profile:

```
zSH> show gpon-olt-config
max-rt-propagation-delay:----->   {0 - 0}
max-onu-response-time:-------->   {0 - 0}
preassigned-eqd:-------------->   {0 - 0}
```

```
los-alpha:--------------------->   {0 - 0}
lof-alpha:--------------------->   {0 - 0}
loam-alpha:-------------------->   {0 - 0}
scrambler:--------------------->   enabled  disabled
fec-mode:---------------------->   enabled  disabled
auto-learn:-------------------->   enabled  disabled
power-level:------------------->   {0 - 0}
guard-bit-count:--------------->   {0 - 0}
dba-mode:---------------------->   predictive  piggyback
wholereport
gem-block-size:---------------->   {0 - 0}
us-ber-interval:--------------->   {0 - 0}
ds-ber-interval:--------------->   {0 - 0}
ber-sf-threshold:-------------->   {3 - 8}
ber-sd-threshold:-------------->   {4 - 9}
fec-request:------------------->   enabled  disabled
key-exchange:------------------>   enabled  disabled
min-rt-propagation-delay:----->   {0 - 0}
min-onu-response-time:-------->   {0 - 0}
eqd-measure-cycles:----------->   {0 - 0}
drift-ctrl-interval:---------->   {0 - 0}
drift-ctrl-limit:------------->   {0 - 0}
alloc-cycle-length:----------->   {1 - 10}
min-us-alloc:----------------->   {0 - 0}
ack-timeout:------------------>   {0 - 0}
pls-max-alloc-size:----------->   {0 - 0}
dba-cycle:--------------------->   {2 - 10}
sr-dba-reporting-block-size:-->   {0 - 0}
protection-switchover-timer:-->   {0 - 0}
preamble-override:------------>   enabled  disabled
preamble-type-0:--------------->   {8}
preamble-type-1:--------------->   {8}
preamble-type-3-pre-range:---->   {8}
preamble-type-3-post-range:--->   {8}
preamble-type-3-pattern:------>   {8}
bip-error-monitoring-mode:---->   disabled  monitoronly
blockonerror
errors-per-sample-threshold:-->   {0 - 0}
errored-samples-threshold:---->   {0 - 0}
bip-max-sample-gap:----------->   {0 - 0}
rogue-onu-detection:---------->   disabled  roguerssi
backgroundprocess
rogue-onu-detect-frequency:--->   {1 - 60}
rogue-onu-rx-power-threshold:->   {0 - 0}
```

Table 33: rogue ONU detection attributes in gpon-olt-config profile

Attribute	Description
rogue-onu-detection	Disable or enable the rogue RSSI detection or background process detection. Values: **disabled** Disable both the rogue RSSI detection and background process detection. **roguerssi** Enable rogue RSSI detection. When a rogue ONU RSSI measurement crosses the rogue-onu-rx-power-threshold, trigger a local alarm and send a trap to ZMS. **backgroundprocess** Enable background process detection. When a rogue transmission is detected, trigger a local alarm and send trap to ZMS. Default: **disabled**
rogue-onu-detect-frequency	How often to run a detection after enabling the detection. Default: **10 seconds**
rogue-onu-rx-power-threshold	Upstream Receive Power High Threshold value for detecting rogue ONU, in the unit of dbm. If the rogue ONT's RSSI measured upstream receive power is higher than the threshold, the RSSI rogue ONU alarm will be reported. Default: **-30**

Configuring Rogue ONU alarm

1 Run the periodical background process.

This example enables the detection mode to **backgroundprocess**, and keeps the default value of detect frequency as 10 seconds.

```
zSH> update gpon-olt-config 1-1-1-0/gponolt
gpon-olt-config  1-1-1-0/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}:
sr-dba-reporting-block-size: -->  {48}:
protection-switchover-timer: -->  {500}:
preamble-override: ------------>  {disabled}:
preamble-type-0: -------------->  {0x00}:
preamble-type-1: -------------->  {0x00}:
preamble-type-3-pre-range: ---->  {0x0b}:
preamble-type-3-post-range: --->  {0x08}:
preamble-type-3-pattern: ------>  {0xaa}:
bip-error-monitoring-mode: ---->  {monitoronly}:
errors-per-sample-threshold: -->  {100}:
errored-samples-threshold: ---->  {10}:
bip-max-sample-gap: ----------->  {10}:
rogue-onu-detection: ---------->
{disabled}:backgroundprocess
rogue-onu-detect-frequency: --->  {10}:
rogue-onu-rx-power-threshold: ->  {-30}:
....................

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

2 View raised alarms. This example shows a rogue ONU is detected.

```
zSH> alarm show
************    Central Alarm Manager    ************
        ActiveAlarmCurrentCount         :8
        AlarmTotalCount                 :15
        ClearAlarmTotalCount            :7
        OverflowAlarmTableCount         :0
ResourceId              AlarmType                                AlarmSeverity
----------              ---------                                -------------
1-1-3-0/eth             linkDown                                    critical
1-1-4-0/gponolt         gpon_olt_rogue_onu_detected               minor
...
```

Configuring RSSI rogue ONU alarm

1 Run the RSSI rogue ONT detection.

This example enables the detection mode to **roguerssi**, keeps the default value of detect frequency as 10 seconds, raises the ONU transceived power threshold from the default value -30 dbm to -10 dbm.

```
zSH> update gpon-olt-config 1-1-1-0/gponolt
gpon-olt-config  1-1-1-0/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}:
sr-dba-reporting-block-size: -->  {48}:
protection-switchover-timer: -->  {500}:
preamble-override: ------------>  {disabled}:
preamble-type-0: -------------->  {0x00}:
preamble-type-1: -------------->  {0x00}:
preamble-type-3-pre-range: ---->  {0x0b}:
preamble-type-3-post-range: --->  {0x08}:
preamble-type-3-pattern: ------>  {0xaa}:
bip-error-monitoring-mode: ---->  {monitoronly}:
errors-per-sample-threshold: -->  {100}:
errored-samples-threshold: ---->  {10}:
bip-max-sample-gap: ----------->  {10}:
rogue-onu-detection: ---------->  {disabled}:roguerssi
rogue-onu-detect-frequency: --->  {10}:
rogue-onu-rx-power-threshold: ->  {-30}: -10
....................

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

2 View raised alarms. This example shows the RSSI measured on a rogue ONU is higher than the threshold.

```
zSH> alarm show
************    Central Alarm Manager    ************
        ActiveAlarmCurrentCount         :8
```

```
        AlarmTotalCount                 :15
        ClearAlarmTotalCount            :7
        OverflowAlarmTableCount         :0
ResourceId               AlarmType                                  AlarmSeverity
----------               ---------                                  -------------
1-1-3-0/eth              linkDown                                      critical
1-1-4-0/gponolt         gpon_olt_rssi_rogue_onu_detected             minor
...
```

12 LINK AGGREGATION CONFIGURATION

This chapter explains how to configure link aggregation for use on the MXK-194/198 FE/GE or 10GE uplink ports:

- Link aggregation overview, page 509
- Configure link aggregation, page 510

Link aggregation overview

This section describes:

- Link aggregation and LACP, page 509
- Link aggregation modes, page 509
- Link resiliency, page 510

The MXK-194/198 supports 802.3ad link aggregation on the FE/GE uplink ports. Link aggregation allows aggregating physical Gigabit Ethernet ports into one single aggregated logical port for additional bandwidth capacity and resiliency.

Link aggregation and LACP

The MXK-194/198 uplink ports 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.

Note: The Ethernet switch on the remote end will need to be configured for link aggregation.

Link aggregation modes

Link aggregation has four modes with the default set to *on*:

- *on*

  This Ethernet link can be aggregated manually using the **linkagg** command. LACP messages are not sent from this port, and any received LACP messages are ignored.

- *active*

  The setting for LACP use, the Ethernet link sends and receives LACP messages and link aggregates automatically when the remote system responds with the appropriate LACP messages.

- *off*

  This Ethernet link cannot be aggregated either manually or dynamically; LACP is not sent from this port and any received LACP messages are ignored.

- *passive*

  This mode sets a link to receive LACP messages, and responds with LACP when receiving a far-end LACP initiation.

Table 34 shows the compatibility matrix for the four settings.

Table 34: LACP compatibility matrix settings

Device one	Device two	Comments
active	*active*	Both devices are sending and receiving LACP. Recommended setting for dynamic aggregation.
active	*passive*	One side of the connection between devices attempts to negotiate a aggregated group. Functional, but not recommended.
on	*on*	Will make links available for manual aggregating; only 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.

Configure link aggregation

This section discusses:

- Configure Ethernet uplink ports for manual link aggregation, page 511
- Configure Ethernet uplink ports for LACP, page 512
- lacp command, page 513
- Configure link aggregation bridges, page 514

Configuring the MXK-194/198 to run link aggregation typically 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 command line interface (CLI).

When the mode is *active*, the mode is changed from *on* to *active* with the **linkagg update link** *interface/type on | active* command. The link aggregation group is automatically created and is composed of up to two or more links depending on the remote device.

Configure Ethernet uplink ports for manual link aggregation

Enter the commands to manually create link aggregation groups on the MXK-194/198 from the CLI. 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 a link aggregation group manually

To create a link aggregation between the two Ethernet uplink ports, use the **linkagg add group** command.

1. To create link aggregation between the two FE/GE uplink logical ports *1-1-2-0/* and *1-1-3-0/* use the **linkagg add group** command.

```
zSH> linkagg add group 1-1-1-0/linkagg link 1-1-2-0/eth
Interface 1-1-1-0/linkagg does not exist
Link aggregation successfully created.
```

Note: Ignore the message:

`Interface 1-1-1-0/linkagg does not exist`

group can be any user-defined string.

Use the same **group** name for both of the ports you are aggregating.

```
zSH> linkagg add group 1-1-1-0/linkagg link 1-1-3-0/eth
Link aggregation successfully created.
```

2. 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
-------------------------------------------------------------------------------
  1   1   1-1-1-0       00:00:00:00:00:00       0x0     0x0      up      2
        links           slot    port    subport         admin
        ---------------------------------------------------------
         1-1-2-0         1       2       0               up
         1-1-3-0         1       3       0               up
```

Deleting a link aggregated group

To delete the link aggregated group:

Enter the **linkagg delete group** command:

```
zSH> linkagg delete group 1-1-1-0/linkagg link 1-1-2-0/eth
Link successfully deleted from aggregation.

zSH> linkagg delete group 1-1-1-0/linkagg link 1-1-3-0/eth
Link successfully deleted from aggregation.
```

Configure Ethernet uplink ports for LACP

When the aggregation mode on the Ethernet uplink ports is set to *active*, the device is able to receive and send LACP and the link aggregation is dynamic, i.e. groups are automatically created. The mode is changed from the default *on* to *active* on the Ethernet uplink ports by entering the **linkagg update link** *interface/type on* | *active* command from the CLI.

Enabling LACP on Ethernet uplink ports

Enable two Ethernet uplink ports for LACP.

1. Connect the MXK-194/198 to the LACP enabled switch.
2. Change the mode from *on* to *active*.

```
zSH> linkagg update link 1-1-5-0/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 1-1-4-0/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
```

If necessary, view the mode in the **aggregationMode** parameter of the **ether** profile.

```
zSH> get ether 1-1-4-0/eth
ether  1-1-4-0/eth
autonegstatus: ---->  {enabled}
mauType: ---------->  {mau1000basetfd}
restart: ---------->  {norestart}
ifType: ----------->  {mau1000basetfd}
autonegcap: ------->
{b10baseT+b10baseTFD+b100baseTX+b100baseTXFD+b1000baseT+b1000baseTFD}
```

```
   remotefault: ------>  {noerror}
   clksrc: ----------->  {automatic}
   pauseFlowControl: ->  {disabled}
   aggregationMode: -->  {active} <-----------
   linkStateMirror: -->  {0/0/0/0/0}
```

lacp command

Use the **lacp** command to verify that the *aggregation partner key* number of the 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-194/198 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 Enabled Traffic Enabled
actor state:3f
partner state:3d
1: partner key 2b67, par port pri 1, partner port # 905, actor state LACP_ACTIVITY
LACP_TIMEOUT AGGREGATION SYNCHRONIZATION COLLECTING DISTRIBUTING , partner state
LACP_ACTIVITY AGGREGATION SYNCHRONIZATION COLLECTING DISTRIBUTING
partner system: 00:0c:db:e8:7e:00
1: agg id 5632180, par sys pri: 1, agg partner key 2b67
par sys: 00:0c:db:e8:7e:00

zSH> lacp monitor 3
PORT 3:
selected = SELECTED Enabled Traffic Enabled
actor state:3d
partner state:3d
2: partner key 2b67, par port pri 1, partner port # 834, actor state LACP_ACTIVITY
AGGREGATION SYNCHRONIZATION COLLECTING DISTRIBUTING , partner state LACP_ACTIVITY
AGGREGATION SYNCHRONIZATION COLLECTING DISTRIBUTING
partner system: 00:0c:db:e8:7e:00
2: agg id 5632180, par sys pri: 1, agg partner key 2b67
par sys: 00:0c:db:e8:7e:00
```

Delete a link aggregation group

Deleting a link aggregation group

Delete each link in the group individually.

1 View the link aggregation group and the links.

```
zSH> linkagg show
LinkAggregations:
slot unit ifName        partner: Sys            Pri     grp ID  admin   numLinks
-------------------------------------------------------------------------------
  1   1   1-1-1-0       00:00:00:00:00:00       0x0     0x0      up      2
        links           slot    port    subport         admin
        ---------------------------------------------------------------
         1-1-2-0         1       2      0               up
         1-1-3-0         1       3      0               up
```

2 Delete the links to delete the link aggregation group.

```
zSH> linkagg delete group 1-1-1-0/linkagg link 1-1-2-0/eth
Link successfully deleted from aggregation.

zSH> linkagg delete group 1-1-1-0/linkagg link 1-1-3-0/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 bridges

Bridges can be added to link aggregation interfaces.

Creating a linkagg intralink bridge

Add an intralink bridge on the logical link aggregation interface.

```
zSH> bridge add 1-1-1-0/linkagg intralink vlan 300
Adding bridge on 1-1-1-0/linkagg
Created bridge-interface-record linkagg-1-1-300/bridge
Bridge-path added successfully
```

The bridge path is automatically created.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
       300 linkagg-1-1-300/bridge              Intralink
```

Adding a bridge to a link aggregated Ethernet port

If a bridge is created on a link aggregated Ethernet interface on a physical port, a *linkagg* bridge is automatically created.

```
zSH> bridge add 1-1-2-0/eth uplink vlan 555
Adding bridge on 1-1-2-0/eth
Created bridge-interface-record linkagg-1-1-555/bridge
Bridge-path added successfully
```

The uplink *linkagg* bridge path is automatically created.

```
zSH> bridge-path show
VLAN/SLAN   Bridge                              Address
-------------------------------------------------------------------------------
      555 linkagg-1-1-555/bridge            Default, Age: 3600, MCAST Age: 150, IGMP Query
Interval: 70, Flap Mode: Default
```

View the link aggregation bridge on the Ethernet port 2.

```
zSH> bridge show
     Orig
Type VLAN/SLAN  VLAN/SLAN      Physical                 Bridge                        St  Table Data
-------------------------------------------------------------------------------
upl            Tagged  555    1/1/1/0/linkagg          linkagg-1-1-555/bridge        DWN S VLAN 555 default
1 Bridge Interfaces displayed
```

13 DIAGNOSTICS AND ADMINISTRATION

This chapter describes tasks you might need to perform to administer the MXK-194/198. It includes the following information:

- IP Service Level Agreement (IPSLA), page 517
- MXK-194/198 logs, page 525
- SNMP, page 535
- SNMP Statistics, page 537
- Bridge statistics, page 538
- Enhanced Ethernet port statistics, page 538
- PON Statistics, page 555
- System maintenance, page 566

IP Service Level Agreement (IPSLA)

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 EHCOREQs/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-194/198 and 4 IPSLA paths per IP device.

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.

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 GPON ONTs, MXK-194/198 ports and other SLMS devices. Figure 52 illustrates IPSLA.

Figure 52: 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}
```

Use **ipsla modify global state** to enable IPSLA and set the polling interval to 120 seconds.

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

2 Create an 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 over a maximum of 32 MXK-194/198 and 4 CPE IPSLA paths per device.

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

```
zSH> ipsla add path ipaddress 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 172.16.78.11 state disabled
```

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

```
zSH> ipsla delete path ipaddress 172.16.78.11
```

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) as shown in Table 35. The following CoS actions are defined by default.

Table 35: CoS action index

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

Table 35: CoS action index (Continued)

Default name	CoS action index	CoS queue
NetwCtrl	8	7

Each CoS action contains the following parameters as described in Table 36:

Table 36: 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 Configured 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. Table 37 shows the default mappings. A COS Action Index of 0 indicates that the DSCP is not used.

Table 37: Default CoS action index mapped to DSCP

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
```

Table 38 explains the statistics for the configured paths.

Table 38: IPSLA statistics for configured paths

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

Table 39 explains the CoS Action Index statistics.

Table 39: CoS action index statistics

CoS action index statistics	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. Calculated as (RTT1 + RTT2 + RTT3 + …….+RTTn)/n where n equals the number of successful ping attempts 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....
```

MXK-194/198 logs

This section provides the following information on how logs work on the MXK-194/198:

- Overview, page 525
- Default log store level, page 526
- Improved user login notification, page 526
- log filter command, page 526
- Enable/disable temporary logging sessions, page 528
- Send logging information to a syslog server, page 528
- Create log modules, page 530
- Log message format, page 532
- Modify log levels, page 533
- Use the log cache, page 534
- View persistent logs, page 535

Overview

Logs enable administrators to monitor system events by generating system messages. It sends these message to:

- A temporary management session (either on the serial craft port or over a telnet session.)
- A syslog server (optional)
- Log modules to create permanent log files

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.

Default log store level

The default log store level is now set to emergency so by default the **log display** command displays only emergency level messages. To enable logging to the consolelog file in the “log” directory, use the **consolelog on** command. If you want to stop the logging to the consolelog files, use the **consolelog off** command.

Use the **log cache** command to display all messages that have been logged to console.

Use the **cd log** and **dir** commands to view the log file history. The log files in this directory record console activity on the MXK-194/198 for the running image, and preserve a copy of the last two reboots. The files *consolelog1.txt* and *consolelog2.txt* hold 10000 lines of console output each. Once the file reaches 10000 lines, the filename is changed to *.old* and a new *.txt* file is used. After a reboot, the *.txt* files are also saved as .old files. Use the **consolelog display <filename>** command to view the contents for a consolelog file. These files are used for troubleshooting and system activity monitoring.

Improved user login notification

Notifications of user login are sent to the console log.

```
SEP 28 10:51:12: alert  : 1/1/1027: clitask0: User admin logged in on slot 1
SEP 28 10:26:39: alert  : 1/1/1028: clitask1: User admin logged out from slot 1
Web Interface for Raptor XP 150A Initialized. User 199.190.211.3
```

log filter command

The **log filter** command allows users to configure custom log levels.

log filter

The **log filter** command enables users to configure custom log levels by:

- ifindex
- slot and port

- VPI and VCI for a VCL
- subscriber endpoint
- Interface group and CID
- ifstack high-ifindex, low-ifindex

Entries are recorded for the specified object and selected log level and all higher levels. Supported log levels from highest to lowest are emergency, alert, critical, error, warning, notice, information, and debug. The wildcards '*' and 'ANY' can be used to manage multiple filters.

Syntax **`log filter set [ifindex ifindex] | [port slot/port] | [vcl ifindex vpi vci] | [subscriber endpoint] | [igcrv ivcrv] | [ifstack ifindex high-ifindex low-ifindex] loglevel`**

ifindex
The desired ifindex number. Use the **get if-translate** command to show ifindex number.

```
zSH> get if-translate 1-1-2-0/eth
if-translate  1-1-2-0/eth
ifIndex: ----------->  {3}
shelf: ------------->  {1}
slot: -------------->  {1}
port: -------------->  {2}
subport: ----------->  {0}
type: -------------->  {eth}
adminstatus: ------->  {up}
physical-flag: ----->  {true}
iftype-extension: -->  {none}
ifName: ------------>  {1-1-2-0}
redundancy-param1: ->  {0}
```

slot port
Slot and port to which the log filter is applied.

vpi vci
Virtual path and channel to which the log filter is applied.

endpoint
Subscriber endpoint to which the log filter is applied.

vpi vci cid
Virtual path, channel, and call ID to which the log filter is applied.

ig crv
Interface group and customer record to which the log filter is applied.

high-ifindex low-ifindex
High and low ifindex values to which the log filter is applied.

loglevel
Sets the level for which log entries are recorded. Entries are recorded for the selected log level and all higher levels. Supported levels from highest

to lowest are emergency, alert, critical, error, warning, notice, information, and debug.

Syntax **log filter show**

Displays all currently defined filters.

Syntax **log filter delete index**

Example

```
zSH> log filter set port 1 1 alert
New filter saved.

zSH> log filter show
Index   Type            Filter Parameters
------  ------------    -----------------------------
  1     Port            slot=1, port=1, logLevel=alert

zSH> log filter delete 1
Log filter 1 deleted
```

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 logging for the current telnet session only:

```
zSH> log session off
Logging disabled.

zSH> log session on
Logging enabled.
```

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:

```
zSH> log serial on
Serial port logging enabled.

zSH> log serial off
Serial port logging disabled.
```

Send logging information to a syslog server

Table 40 describes the parameters in the **syslog-destination** profile to send messages to a syslog server.

Table 40: 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 set 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.
```

Create log modules

The **log-module** command creates log files that will persist across system reboots and power cycles. The **log-module** profile supports the configuration of persistent log messages, syslog messages, and persistent storage levels by module. Modify this profile when you want to send different messages to admin sessions, the persistent logs, and the syslog server.Table 41 describes the **log-module** parameters.

Table 41: 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 41: 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.
```

Log message format

Table 42 describes the information each log message can contain.

Table 42: Default log message fields

Option	Description
Time	Time stamp of log message. Enabled by default.
Date	Date stamp of log message. Enabled by default.
Level	Logging level of the message. Enabled by default.
Taskname	Name of task that generated the log message. This is generally useful only for Zhone development engineers. Enabled by default.
Task ID	Task ID that generated the log message.This is generally useful only for Zhone development engineers.
File	System filename that generated the log message. This is generally useful only for Zhone development engineers.
Function	Function that generated the log message. This is generally useful only for Zhone development engineers.
Line	Line in code that generated the log message. This is generally useful only for Zhone development engineers.

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 >
                   < globalticks   | 14 > < on | off >
                   < all           | 14 > < on | off >
                   < default       | 15 > < on | off >
options 'time' & 'date' supercede option 'ticks'
time: date: level: address: log: port: category: system:  (0x707)
```

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)
```

Modify log levels

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

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

```
zSH> log show
MODULE                                                  LEVEL          STATUS
alarm_mgr                                               error          enabled
bds                                                     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
cli                                                     error          enabled
cssagent                                                error          enabled
dhcpclient                                              error          enabled
    dhcpclib                                            error          enabled
dhcpmibhandler                                          error          enabled
dhcpsax                                                 error          enabled
dhcpserver                                              error          enabled
diags                                                   error          enabled
.....
```

Log levels determine the number of messages that are displayed on the console. The higher the log level, the more messages are displayed. The MXK-194/198 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 level** *module* command. For example, the following command changes the card module logging level to emergency:

```
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
```

Use the log cache

The **log cache** command displays the non-persistent log messages. It uses the following syntax:

log cache

Displays the log cache.

log cache max length

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 grep pattern

Searches through the log cache for the specified regular expression.

log cache clear

Clears the log cache.

log cache size

Sets the maximum amount of memory for the log cache. Without options, displays the current log size.

log cache help

Displays help on the **log cache** command.

Examples

To change the current configured log cache size:

```
zSH> log cache max 200
Maximum number of log messages that can be saved: 200
```

The following example searches through the log cache for the string “Major”:

```
zSH> log cache grep Major
Searching for: "Major"
[1]: FEB 07 11:18:42: alert  : 1/1/1025: alarm_mgr:
tLineAlarm: 01:01:01 Major D
S1 Down Line 1:1:1:0 (FarEnd Rx LOF)[2]: FEB 07 11:18:42:
alert  : 1/1/1025: alarm_mgr: tLineAlarm: 01:01:02 Major
D
```

```
S1 Down Line 1:1:2:0 (FarEnd Rx LOF)[3]: FEB 07 11:18:42:
alert  : 1/1/1025: alarm_mgr: tLineAlarm: 01:01:03 Major
D
S1 Down Line 1:1:3:0 (FarEnd Rx LOF)
...
...
...
```

View persistent logs

Use the **log display** command to view the persistent logs.

SNMP

This section describes the following:

- Create SNMP community names and access lists, page 535
- Configure traps, page 537

Create SNMP community names and access lists

Note: By default, the MXK-194/198 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 lists 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 lists

The following example defines a community name **private** with read/write privileges and also creates an access list to verify that the SNMP managers attempting to access the MXK-194/198 are 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-194/198 will send traps to. To configure a trap destination you need to know:

- the IP address of the SNMP manager workstation
- the community name the trap recipient expects

Note that the **resendseqno** and **ackedseqno** parameters are set by the ZMS. The other parameters in the **trap-destination** profile can be left at their default values. The source IP used in all MXK-194/198 generated traps is the ipaddress defined in the **system 0** profile. 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}:
gatewaytrapserveraddr:->  {36}:
...................
Save new record? [s]ave, [c]hange or [q]uit: s
New record saved.
```

SNMP Statistics

The MXK-194/198 supports the following SNMP statistics

- MIB II statistics:
  - ifHCIn/OutOctets
  - ifHCIn/OutUCastPkts
  - ifHCIn/OutMultiCastPkts
  - ifHCIn/OutBroadCastPkts
  - ifInDiscards
  - ifOutDiscards
  - ifInErrors

  - ifOutErrors
  - ifOperStatus
  - ifLastChange
- Ethernet statistics:
  - A transmitted frame inhibited by a single collision
  - A transmitted frame inhibited by multiple collisions
  - Frames received that exceed the maximum frame size
  - Frames received that failed the FCS check
- OLT statistics
- ONU statistics

Bridge statistics

The **bridge stats** command displays packet information on all configured bridges.

```
zSH> bridge stats
Interface                           Received Packets        Transmitted Packets
Name                               UCast  MCast  BCast   UCast  MCast  Bcast  Error
1-1-2-901-gponport-998/bridge      --     --     --      0      40     0      0
1-1-2-902-gponport-998/bridge      --     --     --      0      40     0      0
1-1-2-903-gponport-998/bridge      --     --     --      0      40     0      0
1-1-1-732-gponport-998/bridge      --     --     --      0      39     0      0
1-1-9-0-eth-840/bridge             --     --     --      --     --     --     --
5 Bridge Interfaces displayed
```

The **bridge stats** *interface* command displays packet information for the designated interface.

```
zSH> bridge stats 1-1-9-0-eth-840/bridge
Interface                           Received Packets        Transmitted Packets
Name                               UCast  MCast  BCast   UCast  MCast  Bcast  Error
1-1-9-0-eth-840/bridge             --     --     --      --     --     --     --
1 Bridge Interfaces displayed

zSH> bridge stats 1-1-2-901-gponport-998/bridge
Interface                           Received Packets        Transmitted Packets
Name                               UCast  MCast  BCast   UCast  MCast  Bcast  Error
1-1-2-901-gponport-998/bridge      --     --     --      0      44     0      0
1 Bridge Interfaces displayed
```

Enhanced Ethernet port statistics

Use **port stats** command to display or clear various statistical information.

port stats *<ifName/Type> <intf|rmon|eth|all|clear>*

The **port stats** *interface/type intf* command displays mib2 interface statistics.

See Table 43 on page 542 for parameter definitions.

```
zSH> port stats 1-1-3-0/eth intf
Interface Name                                          1-1-3-0
Operational Status                                      Up
Received Bytes                                          0
Received Packets                                        0
Received Multicast Packets                              0
Received Broadcast Packets                              0
Transmitted Bytes                                       45056
Transmitted Unicast Packets                             0
Transmitted Multicast Packets                           2
Transmitted Broadcast Packets                           702
Received Discards                                       0
Received Errors                                         0
Received Unknown Protocols                              0
Transmitted Discards                                    0
Transmitted Errors                                      0
Speed Bits per Second                                   *** n/a ***
Speed Megabits per Second                               1000
```

The **port stats** *interface/type rmon* command displays Ethernet remote monitoring statistics.

```
zSH> port stats 1-1-3-0/eth rmon
Total Dropped Events                                    0
Total Dropped Frames                                    0
Total Bytes                                             45696
Total Packets                                           714
Transmitted Packets                                     714
Received Packets                                        0
Transmitted Multicast Bytes                             0
Received Multicast Bytes                                0
Received Multicast Dropped Bytes                        0
Transmitted Average Throughput                          24
Received Average Throughput                             0
Transmitted Bandwidth Occupancy                         0
Received Bandwidth Occupancy                            0
Total Broadcast Packets                                 712
Total Multicast Packets                                 2
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                                   714
Received No Errors                                      0
IPMC Bridged Packets                                    0
IPMC Routed Packets                                     0
Transmitted IPMC Dropped Packets                        0
```

```
Received IPMC Dropped Packets                                0
Total Packets 0 to 64 Bytes                                  714
Total Packets 65 to 127 Bytes                                0
Total Packets 128 to 255 Bytes                               0
Total Packets 256 to 511 Bytes                               0
Total Packets 512 to 1023 Bytes                              0
Total Packets 1024 to 1518 Bytes                             0
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                               0
Received Packets 65 to 127 Bytes                             0
Received Packets 128 to 255 Bytes                            0
Received Packets 256 to 511 Bytes                            0
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                            714
Transmitted Packets 65 to 127 Bytes                          0
Transmitted Packets 128 to 255 Bytes                         0
Transmitted Packets 256 to 511 Bytes                         0
Transmitted Packets 512 to 1023 Bytes                        0
Transmitted Packets 1024 to 1518 Bytes                       0
Transmitted Packets 1519 to 2047 Bytes                       0
Transmitted Packets 2048 to 4095 Bytes                       0
Transmitted Packets 4095 to 9216 Bytes                       0
```

The **port stats** *interface/type eth* command displays the Ethernet dot3 statistics.

```
zSH> port stats 1-1-3-0/eth eth
Alignment Errors                                             0
FCS Errors                                                   0
Single Collision Frames                                      0
Multiple Collision Frames                                    0
SQE Test Errors                                              0
Deferred Transmissions                                       0
Late Collisions                                              0
Excessive Collisions                                         0
Internal Mac Transmit Errors                                 0
Carrier Sense Errors                                         0
FrameTooLongs                                                0
InternalMacReceiveErrors                                     0
SymbolErrors                                                 0
DuplexStatus                                                 Full
```

The **port stats** *interface/type all* commands displays all of the Ethernet statistics.

```
zSH> port stats 1-1-3-0/eth all
******  eth  ******
Alignment Errors                                             0
```

```
FCS Errors                                          0
Single Collision Frames                             0
Multiple Collision Frames                           0
SQE Test Errors                                     0
Deferred Transmissions                              0
Late Collisions                                     0
Excessive Collisions                                0
Internal Mac Transmit Errors                        0
Carrier Sense Errors                                0
FrameTooLongs                                       0
InternalMacReceiveErrors                            0
SymbolErrors                                        0
DuplexStatus                                        Full
******  rmon  ******
Total Dropped Events                                0
Total Dropped Frames                                0
Total Bytes                                         47360
Total Packets                                       740
Transmitted Packets                                 740
Received Packets                                    0
Transmitted Multicast Bytes                         0
Received Multicast Bytes                            0
Received Multicast Dropped Bytes                    0
Transmitted Average Throughput                      32
Received Average Throughput                         0
Transmitted Bandwidth Occupancy                     0
Received Bandwidth Occupancy                        0
Total Broadcast Packets                             738
Total Multicast Packets                             2
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                               740
Received No Errors                                  0
IPMC Bridged Packets                                0
IPMC Routed Packets                                 0
Transmitted IPMC Dropped Packets                    0
Received IPMC Dropped Packets                       0
Total Packets 0 to 64 Bytes                         740
Total Packets 65 to 127 Bytes                       0
Total Packets 128 to 255 Bytes                      0
Total Packets 256 to 511 Bytes                      0
Total Packets 512 to 1023 Bytes                     0
Total Packets 1024 to 1518 Bytes                    0
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                      0
Received Packets 65 to 127 Bytes                    0
```

```
Received Packets 128 to 255 Bytes                          0
Received Packets 256 to 511 Bytes                          0
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                          740
Transmitted Packets 65 to 127 Bytes                        0
Transmitted Packets 128 to 255 Bytes                       0
Transmitted Packets 256 to 511 Bytes                       0
Transmitted Packets 512 to 1023 Bytes                      0
Transmitted Packets 1024 to 1518 Bytes                     0
Transmitted Packets 1519 to 2047 Bytes                     0
Transmitted Packets 2048 to 4095 Bytes                     0
Transmitted Packets 4095 to 9216 Bytes                     0
******  intf  ******
Interface Name                                             1-1-3-0
Operational Status                                         Up
Received Bytes                                             0
Received Packets                                           0
Received Multicast Packets                                 0
Received Broadcast Packets                                 0
Transmitted Bytes                                          47360
Transmitted Unicast Packets                                0
Transmitted Multicast Packets                              2
Transmitted Broadcast Packets                              738
Received Discards                                          0
Received Errors                                            0
Received Unknown Protocols                                 0
Transmitted Discards                                       0
Transmitted Errors                                         0
Speed Bits per Second                                      *** n/a ***
Speed Megabits per Second                                  1000
```

The **port stats clear** *interface/type* command clears all port stats counters.

```
zSH> port stats clear 1-1-3-0/eth
INTF Stats cleared
```

Table 43 defines the parameters for all of the Ethernet statistics.

Table 43: MXK-194/198 Enhanced Ethernet port statistics

Parameter	Description
eth	

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Alignment Errors	A count of frames received on a particular interface that are not an integral number of octets in length and do not pass the FCS check. The count represented by an instance of this object is incremented when the alignment Error status is returned by the MAC service to the LLC (or other MAC user). Received frames for which multiple error conditions obtain are, according to the conventions of IEEE 802.3 Layer Management, counted exclusively according to the error status presented to the LLC. This counter does not increment for 8-bit wide group encoding schemes. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
FCS Errors	A count of frames received on a particular interface that are an integral number of octets in length but do not pass the FCS check. This count does not include frames received with frame-too-long or frame-too-short error. The count represented by an instance of this object is incremented when the frameCheckError status is returned by the MAC service to the LLC (or other MAC user). Received frames for which multiple error conditions obtain are, according to the conventions of IEEE 802.3 Layer Management, counted exclusively according to the error status presented to the LLC. Note: Coding errors detected by the physical layer for speeds above 10 Mb/s will cause the frame to fail the FCS check. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Single Collision Frames	A count of successfully transmitted frames on a particular interface for which transmission is inhibited by exactly one collision. A frame that is counted by an instance of this object is also counted by the corresponding instance of either the ifOutUcastPkts, ifOutMulticastPkts, or ifOutBroadcastPkts, and is not counted by the corresponding instance of the dot3StatsMultipleCollisionFrames object. This counter does not increment when the interface is operating in full-duplex mode. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Multiple Collision Frames	A count of successfully transmitted frames on a particular interface for which transmission is inhibited by more than one collision. A frame that is counted by an instance of this object is also counted by the corresponding instance of either the ifOutUcastPkts, ifOutMulticastPkts, or ifOutBroadcastPkts, and is not counted by the corresponding instance of the dot3StatsSingleCollisionFrames object. This counter does not increment when the interface is operating in full-duplex mode. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
SQE Test Errors	A count of times that the SQE TEST ERROR message is generated by the PLS sublayer for a particular interface. The SQE TEST ERROR is set in accordance with the rules for verification of the SQE detection mechanism in the PLS Carrier Sense Function as described in IEEE Std. 802.3, 1998 Edition, section 7.2.4.6. This counter does not increment on interfaces operating at speeds greater than 10 Mb/s, or on interfaces operating in full-duplex mode. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Deferred Transmissions	A count of frames for which the first transmission attempt on a particular interface is delayed because the medium is busy. The count represented by an instance of this object does not include frames involved in collisions. This counter does not increment when the interface is operating in full-duplex mode. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Late Collisions	The number of times that a collision is detected on a particular interface later than one slotTime into the transmission of a packet. A (late) collision included in a count represented by an instance of this object is also considered as a (generic) collision for purposes of other collision-related statistics. This counter does not increment when the interface is operating in full-duplex mode. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Excessive Collisions	A count of frames for which transmission on a particular interface fails due to excessive collisions. This counter does not increment when the interface is operating in full-duplex mode. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Internal Mac Transmit Errors	A count of frames for which transmission on a particular interface fails due to an internal MAC sublayer transmit error. A frame is only counted by an instance of this object if it is not counted by the corresponding instance of either the dot3StatsLateCollisions object, the dot3StatsExcessiveCollisions object, or the dot3StatsCarrierSenseErrors object. The precise meaning of the count represented by an instance of this object is implementation- specific. In particular, an instance of this object may represent a count of transmission errors on a particular interface that are not otherwise counted. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Carrier Sense Errors	The number of times that the carrier sense condition was lost or never asserted when attempting to transmit a frame on a particular interface. The count represented by an instance of this object is incremented at most once per transmission attempt, even if the carrier sense condition fluctuates during a transmission attempt. This counter does not increment when the interface is operating in full-duplex mode. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
FrameTooLongs	A count of frames received on a particular interface that exceed the maximum permitted frame size. The count represented by an instance of this object is incremented when the frameTooLong status is returned by the MAC service to the LLC (or other MAC user). Received frames for which multiple error conditions obtain are, according to the conventions of IEEE 802.3 Layer Management, counted exclusively according to the error status presented to the LLC. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
InternalMacReceive Errors	A count of frames for which reception on a particular interface fails due to an internal MAC sublayer receive error. A frame is only counted by an instance of this object if it is not counted by the corresponding instance of either the dot3StatsFrameTooLongs object, the dot3StatsAlignmentErrors object, or the dot3StatsFCSErrors object. The precise meaning of the count represented by an instance of this object is implementation- specific. In particular, an instance of this object may represent a count of receive errors on a particular interface that are not otherwise counted. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime
SymbolErrors	For an interface operating at 100 Mb/s, the number of times there was an invalid data symbol when a valid carrier was present. For an interface operating in half-duplex mode at 1000 Mb/s, the number of times the receiving media is non-idle (a carrier event) for a period of time equal to or greater than slotTime, and during which there was at least one occurrence of an event that causes the PHY to indicate 'Data reception error' or 'carrier extend error' on the GMII. For an interface operating in full-duplex mode at 1000 Mb/s, the number of times the receiving media is non-idle a carrier event) for a period of time equal to or greater than minFrameSize, and during which there was at least one occurrence of an event that causes the PHY to indicate 'Data reception error' on the GMII. The count represented by an instance of this object is incremented at most once per carrier event, even if multiple symbol errors occur during the carrier event. This count does not increment if a collision is present. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
DuplexStatus	The current mode of operation of the MAC entity. 'unknown' indicates that the current duplex mode could not be determined. Management control of the duplex mode is accomplished through the MAU MIB. When an interface does not support autonegotiation, or when autonegotiation is not enabled, the duplex mode is controlled using ifMauDefaultType. When autonegotiation is supported and enabled, duplex mode is controlled using ifMauAutoNegAdvertisedBits. In either case, the currently operating duplex mode is reflected both in this object and in ifMauType. Note that this object provides redundant information with ifMauType. Normally, redundant objects are discouraged. However, in this instance, it allows a management application to determine the duplex status of an interface without having to know every possible value of ifMauType. This was felt to be sufficiently valuable to justify the redundancy. Values: **unknown** **halfDuplex** **fullDuplex**
rmon	**Remote Network Monitoring**
Total Dropped Events	The total number of events in which packets were dropped by the probe due to lack of resources. Note that this number is not necessarily the number of packets dropped; it is just the number of times this condition has been detected.
Total Dropped Frames	The total number of frames that were received by the probe and therefore not accounted for in the zhoneEtherStatsDropEvents, but that the probe chose not to count for this entry for whatever reason. Most often, this event occurs when the probe is out of some resources and decides to shed load from this collection. This count does not include packets that were not counted because they had MAC-layer errors. Note that, unlike the dropEvents counter, this number is the exact number of frames dropped.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Total Bytes	The total number of octets of data (including those in bad packets) transmitted and received on the network (excluding framing bits but including FCS octets). This object can be used as a reasonable estimate of 10-Megabit ethernet utilization. If greater precision is desired, the zhoneEtherStatsPkts and zhoneEtherStatsOctets objects should be sampled before and after a common interval. The differences in the sampled values are Pkts and Octets, respectively, and the number of seconds in the interval is Interval. These values are used to calculate the Utilization as follows: Pkts * (9.6 + 6.4) + (Octets *.8) Utilization = ---------- Interval * 10,000 The result of this equation is the value Utilization which is the percent utilization of the ethernet segment on a scale of 0 to 100 percent.
Total Packets	The total number of packets (including bad packets, broadcast packets, and multicast packets) transmitted and received.
Transmitted Packets	The total number of packets (including bad packets, broadcast packets, and multicast packets) transmitted.
Received Packets	The total number of packets (including bad packets, broadcast packets, and multicast packets) received.
Transmitted Multicast Bytes	Transmitted multicast bytes.
Received Multicast Bytes	Received multicast bytes.
Received Multicast Dropped Bytes	Dropped multicast bytes.
Transmitted Average Throughput	Average transmit throughput in bits per second since last query. For accuracy purposes, it is recommended that this object be queried in intervals of five (5) seconds or greater.
Received Average Throughput	Average receive throughput in bits per second since last query. For accuracy purposes, it is recommended that this object be queried in intervals of five (5) seconds or greater.
Transmitted Bandwidth Occupancy	Percentage of bandwidth currently being utilized for transmitting traffic. This rate is calculated based on the delta between prior and current query of this object. For accuracy purposes, it is recommended that this object be queried in intervals of five (5) seconds or greater.
Received Bandwidth Occupancy	Percentage of bandwidth currently being utilized for receiving traffic. This rate is calculated based on the delta between prior and current query of this object. For accuracy purposes, it is recommended that this object be queried in intervals of five (5) seconds or greater.
Total Broadcast Packets	The total number of good packets transmitted and received that were directed to the broadcast address. Note that this does not include multicast packets.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Total Multicast Packets	The total number of good packets transmitted and received that were directed to a multicast address. Note that this number does not include packets directed to the broadcast address.
CRC Align Errors	The total number of packets received that had a length (excluding framing bits, but including FCS octets) of between 64 and 1518 octets, inclusive, but had either a bad Frame Check Sequence (FCS) with an integral number of octets (FCS Error) or a bad FCS with a non-integral number of octets (Alignment Error).
Undersize Packets	The total number of packets received that were less than 64 octets long (excluding framing bits, but including FCS octets) and were otherwise well formed.
Oversize Packets	The total number of packets transmitted and received that were longer than 1518 octets (excluding framing bits, but including FCS octets) and were otherwise well formed.
Transmitted Oversize Packets	The total number of packets transmitted that were longer than 1518 octets (excluding framing bits, but including FCS octets) and were otherwise well formed.
Received Oversize Packets	The total number of packets received that were longer than 1518 octets (excluding framing bits, but including FCS octets) and were otherwise well formed.
Fragments	The total number of packets received that were less than 64 octets in length (excluding framing bits but including FCS octets) and had either a bad Frame Check Sequence (FCS) with an integral number of octets (FCS Error) or a bad FCS with a non-integral number of octets (Alignment Error). Note that it is entirely normal for zhoneEtherStatsFragments to increment. This is because it counts both runts (which are normal occurrences due to collisions) and noise hits.
Jabbers	The total number of packets received that were longer than 1518 octets (excluding framing bits, but including FCS octets), and had either a bad Frame Check Sequence (FCS) with an integral number of octets (FCS Error) or a bad FCS with a non-integral number of octets (Alignment Error). Note that this definition of jabber is different than the definition in IEEE-802.3 section 8.2.1.5 (10BASE5) and section 10.3.1.4 (10BASE2). These documents define jabber as the condition where any packet exceeds 20 ms. The allowed range to detect jabber is between 20 ms and 150 ms.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Collisions	The best estimate of the total number of collisions on this Ethernet segment. The value returned will depend on the location of the RMON probe. Section 8.2.1.3 (10BASE-5) and section 10.3.1.3 (10BASE-2) of IEEE standard 802.3 states that a station must detect a collision, in the receive mode, if three or more stations are transmitting simultaneously. A repeater port must detect a collision when two or more stations are transmitting simultaneously. Thus a probe placed on a repeater port could record more collisions than a probe connected to a station on the same segment would. Probe location plays a much smaller role when considering 10BASE-T. 14.2.1.4 (10BASE-T) of IEEE standard 802.3 defines a collision as the simultaneous presence of signals on the DO and RD circuits (transmitting and receiving at the same time). A 10BASE-T station can only detect collisions when it is transmitting. Thus probes placed on a station and a repeater, should report the same number of collisions. Note also that an RMON probe inside a repeater should ideally report collisions between the repeater and one or more other hosts (transmit collisions as defined by IEEE 802.3k) plus receiver collisions observed on any coax segments to which the repeater is connected.
Transmitted No Errors	The total number of TX packets transmitted without error.
Received No Errors	The total number of RX packets received without error.
IPMC Bridged Packets	Broadcom IPMC Bridged Packet count.
IPMC Routed Packets	Broadcom IPMC Routed Packet count.
Transmitted IPMC Dropped Packets	Broadcom IPMC Tx Dropped Packet count.
Received IPMC Dropped Packets	Broadcom IPMC Rx Dropped Packet count.
Total Packets 0 to 64 Bytes	The total number of packets (including bad packets) transmitted and received that were 64 octets in length (excluding framing bits but including FCS octets).
Total Packets 65 to 127 Bytes	The total number of packets (including bad packets) transmitted and received that were between 65 and 127 octets in length inclusive (excluding framing bits but including FCS octets).
Total Packets 128 to 255 Bytes	The total number of packets (including bad packets) transmitted and received that were between 128 and 255 octets in length inclusive (excluding framing bits but including FCS octets).
Total Packets 256 to 511 Bytes	The total number of packets (including bad packets) transmitted and received that were between 256 and 511 octets in length inclusive (excluding framing bits but including FCS octets).
Total Packets 512 to 1023 Bytes	The total number of packets (including bad packets) transmitted and received that were between 512 and 1023 octets in length inclusive (excluding framing bits but including FCS octets).

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Total Packets 1024 to 1518 Bytes	The total number of packets (including bad packets) transmitted and received that were between 1024 and 1518 octets in length inclusive (excluding framing bits but including FCS octets).
Total Packets 1519 to 2047 Bytes	The total number of packets (including bad packets) transmitted and received that were between 1519 and 2047 octets in length inclusive (excluding framing bits but including FCS octets).
Total Packets 2048 to 4095 Bytes	The total number of packets (including bad packets) transmitted and received that were between 2048 and 4095 octets in length inclusive (excluding framing bits but including FCS octets).
Total Packets 4095 to 9216 Bytes	The total number of packets (including bad packets) transmitted and received that were between 4095 and 9216 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 0 to 64 Bytes	The total number of packets (including bad packets) received that were between 0 and 64 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 65 to 127 Bytes	The total number of packets (including bad packets) received that were between 65 and 127 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 128 to 255 Bytes	The total number of packets (including bad packets) received that were between 128 and 255 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 256 to 511 Bytes	The total number of packets (including bad packets) received that were between 256 and 511 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 512 to 1023 Bytes	The total number of packets (including bad packets) received that were between 512 and 1023 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 1024 to 1518 Bytes	The total number of packets (including bad packets) received that were between 1024 and 1518 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 1519 to 2047 Bytes	The total number of packets (including bad packets) received that were between 1519 and 2047 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 2048 to 4095 Bytes	The total number of packets (including bad packets) received that were between 2048 and 4095 octets in length inclusive (excluding framing bits but including FCS octets).
Received Packets 4095 to 9216 Bytes	The total number of packets (including bad packets) received that were between 4095 and 9216 octets in length inclusive (excluding framing bits but including FCS octets).
Transmitted Packets 0 to 64 Bytes	The total number of packets (including bad packets) transmitted that were between 0 and 64 octets in length inclusive (excluding framing bits but including FCS octets).
Transmitted Packets 65 to 127 Bytes	The total number of packets (including bad packets) transmitted that were between 65 and 127 octets in length inclusive (excluding framing bits but including FCS octets).
Transmitted Packets 128 to 255 Bytes	The total number of packets (including bad packets) transmitted that were between 128 and 255 octets in length inclusive (excluding framing bits but including FCS octets).

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Transmitted Packets 256 to 511 Bytes	The total number of packets (including bad packets) transmitted that were between 256 and 511 octets in length inclusive (excluding framing bits but including FCS octets).
Transmitted Packets 512 to 1023 Bytes	The total number of packets (including bad packets) transmitted that were between 512 and 1023 octets in length inclusive (excluding framing bits but including FCS octets).
Transmitted Packets 1024 to 1518 Bytes	The total number of packets (including bad packets) transmitted that were between 1024 and 1518 octets in length inclusive (excluding framing bits but including FCS octets).
Transmitted Packets 1519 to 2047 Bytes	The total number of packets (including bad packets) transmitted that were between 1519 and 2047 octets in length inclusive (excluding framing bits but including FCS octets).
Transmitted Packets 2048 to 4095 Bytes	The total number of packets (including bad packets) transmitted that were between 2048 and 4095 octets in length inclusive (excluding framing bits but including FCS octets).
Transmitted Packets 4095 to 9216 Bytes	The total number of packets (including bad packets) transmitted that were between 4095 and 9216 octets in length inclusive (excluding framing bits but including FCS octets).

intf	**Interface statistics**

Interface Name	The textual name of the interface. The value of this object should be the name of the interface as assigned by the local device and should be suitable for use in commands entered at the device's `console'. This might be a text name, such as `le0' or a simple port number, such as `1', depending on the interface naming syntax of the device. If several entries in the ifTable together represent a single interface as named by the device, then each will have the same value of ifName. Note that for an agent which responds to SNMP queries concerning an interface on some other (proxied) device, then the value of ifName for such an interface is the proxied device's local name for it. If there is no local name, or this object is otherwise not applicable, then this object contains a zero-length string.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Operational Status	The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed. If ifAdminStatus is down(2) then ifOperStatus should be down(2). If ifAdminStatus is changed to up(1) then ifOperStatus should change to up(1) if the interface is ready to transmit and receive network traffic; it should change to dormant(5) if the interface is waiting for external actions (such as a serial line waiting for an incoming connection); it should remain in the down(2) state if and only if there is a fault that prevents it from going to the up(1) state; it should remain in the notPresent(6) state if the interface has missing (typically, hardware) components. Values: **up** **down** **testing** **unknown** **dormant** **notPResent** **lowerLayerDown**
Received Bytes	The total number of octets received on the interface, including framing characters. This object is a 64-bit version of ifInOctets. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value ofifCounterDiscontinuityTime.
Received Multicast Packets	The number of packets, delivered by this sub-layer to a higher (sub-)layer, which were addressed to a multicast address at this sub-layer. For a MAC layer protocol, this includes both Group and Functional addresses. This object is a 64-bit version of ifInMulticastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Received Broadcast Packets	The number of packets, delivered by this sub-layer to a higher (sub-)layer, which were addressed to a broadcast address at this sub-layer. This object is a 64-bit version of ifInBroadcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Transmitted Bytes	The total number of octets transmitted out of the interface, including framing characters. This object is a 64-bit version of ifOutOctets. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Transmitted Unicast Packets	The total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent. This object is a 64-bit version of ifOutUcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Transmitted Multicast Packets	The total number of packets that higher-level protocols requested be transmitted, and which were addressed to a multicast address at this sub-layer, including those that were discarded or not sent. For a MAC layer protocol, this includes both Group and Functional addresses. This object is a 64-bit version of ifOutBroadcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Transmitted Broadcast Packets	The total number of packets that higher-level protocols requested be transmitted, and which were addressed to a broadcast address at this sub-layer, including those that were discarded or not sent. This object is a 64-bit version of ifOutBroadcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Received Discards	The number of inbound packets which were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Received Errors	For packet-oriented interfaces, the number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol. For character-oriented or fixed-length interfaces, the number of inbound transmission units that contained errors preventing them from being deliverable to a higher-layer protocol. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Received Unknown Protocols	For packet-oriented interfaces, the number of packets received via the interface which were discarded because of an unknown or unsupported protocol. For character-oriented or fixed-length interfaces that support protocol multiplexing the number of transmission units received via the interface which were discarded because of an unknown or unsupported protocol. For any interface that does not support protocol multiplexing, this counter will always be 0. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

Table 43: MXK-194/198 Enhanced Ethernet port statistics (Continued)

Parameter	Description
Transmitted Discards	The number of outbound packets which were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Transmitted Errors	For packet-oriented interfaces, the number of outbound packets that could not be transmitted because of errors. For character-oriented or fixed-length interfaces, the number of outbound transmission units that could not be transmitted because of errors. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
Speed Bits per Second	An estimate of the interface's current bandwidth in bits per second. For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth. If the bandwidth of the interface is greater than the maximum value reportable by this object then this object should report its maximum value (4,294,967,295) and ifHighSpeed must be used to report the interace's speed. For a sub-layer which has no concept of bandwidth, this object should be zero.
Speed Megabits per Second	An estimate of the interface's current bandwidth in units of 1,000,000 bits per second. If this object reports a value of `n' then the speed of the interface is somewhere in the range of `n-500,000' to `n+499,999'. For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth. For a sub-layer which has no concept of bandwidth, this object should be zero.

PON Statistics

This section describes the PON statistics.

PON statistics are the OLT or ONU statistics collected and reported by MXK-194/198 OLT.

The Downstream stats are the stats that were sent from MXK-194/198 to ONU, and the upstream stats was sent from ONU to MXK-194/198.

- View OLT statistics, page 555
- View ONU statistics, page 564

View OLT statistics

MXK-194/198 OLT can report these stats types for OLT interfaces: 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 interfaces.

MXK-194/198 OLT can report these stats types for ONU interfaces: 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 interfaces.

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 interfaces.

- **`eth`**

  refers to ethernet dot3 statistics.

- **`olt`**

  refers to GPON OLT traffic management statistics. It is valid for **gponolt** interfaces only.

- **`onu`**

  refers to GPON ONU error statistics as reported by MXK-194/198 OLT. It is valid for **gpononu** interfaces 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 874

if-translate  1-1-1-0/rs232
if-translate  1-1-1-0/eth
if-translate  1-1-1-0-eth/linegroup
if-translate  1-1-2-0/eth
if-translate  1-1-2-0-eth/linegroup
if-translate  1-1-3-0/eth
if-translate  1-1-3-0-eth/linegroup
if-translate  1-1-4-0/eth
if-translate  1-1-4-0-eth/linegroup
if-translate  1-1-5-0/eth
if-translate  1-1-5-0-eth/linegroup
if-translate  1-1-6-0/eth
if-translate  1-1-6-0-eth/linegroup
if-translate  1-1-7-0/eth
if-translate  1-1-7-0-eth/linegroup
if-translate  1-1-8-0/eth
if-translate  1-1-8-0-eth/linegroup
if-translate  1-1-9-0/eth
if-translate  1-1-9-0-eth/linegroup
if-translate  1-1-6-0/ipobridge
if-translate  ipobridge/linegroup
if-translate  1-1-1-0/gponolt
if-translate  1-1-1-0-gponolt/linegroup
if-translate  1-1-2-0/gponolt
if-translate  1-1-2-0-gponolt/linegroup
if-translate  1-1-3-0/gponolt
if-translate  1-1-3-0-gponolt/linegroup
if-translate  1-1-4-0/gponolt
if-translate  1-1-5-0/gponolt
if-translate  1-1-5-0-gponolt/linegroup
```

```
if-translate  1-1-6-0/gponolt
if-translate  1-1-7-0/gponolt
if-translate  1-1-7-0-gponolt/linegroup
if-translate  1-1-8-0/gponolt
if-translate  1-1-8-0-gponolt/linegroup
if-translate  1-1-1-1/gpononu
if-translate  1-1-1-2/gpononu
...
874 entries found.
```

2 When specifying **all** as the stats type, the rmon, OLT and interface stats are displayed for this OLT interface.

```
zSH> port stats 1-1-3-0/gponolt all
******  rmon  ******
Total Dropped Events                                    0
Total Dropped Frames                                    0
Total Bytes                                             0
Total Packets                                           0
Transmitted Packets                                     0
Received Packets                                        0
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                                 0
Total Multicast Packets                                 0
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                                   0
Received No Errors                                      0
IPMC Bridged Packets                                    0
IPMC Routed Packets                                     0
Transmitted IPMC Dropped Packets                        0
Received IPMC Dropped Packets                           0
Total Packets 0 to 64 Bytes                             0
Total Packets 65 to 127 Bytes                           0
Total Packets 128 to 255 Bytes                          0
Total Packets 256 to 511 Bytes                          0
Total Packets 512 to 1023 Bytes                         0
Total Packets 1024 to 1518 Bytes                        0
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                          0
```

```
Received Packets 65 to 127 Bytes                          0
Received Packets 128 to 255 Bytes                         0
Received Packets 256 to 511 Bytes                         0
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                         0
Transmitted Packets 65 to 127 Bytes                       0
Transmitted Packets 128 to 255 Bytes                      0
Transmitted Packets 256 to 511 Bytes                      0
Transmitted Packets 512 to 1023 Bytes                     0
Transmitted Packets 1024 to 1518 Bytes                    0
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                                 0
Upstream Discarded Frames                                 0
Upstream Gem Frames                                       0
Upstream Omci Frames                                      0
Upstream Ploam Frames                                     0
Upstream Idle Ploam Frames                                0
Downstream Valid Gem Frames                               0
Downstream Discarded Frames                               0
Downstream Gem Frames                                     0
Downstream Omci Frames                                    0
Downstream Ploam Frames                                   0
Downstream Idle Ploam Frames                              0
Downstream Pon Valid Ethernet Packtes                     0
Downstream Pon Cpu Packets                                0
Downstream Transmitted Bytes                              0
Upstream Pon Valid Packets                                0
Upstream Pon Valid Not Idle Ploams                        0
Upstream Pon Error Ploams                                 0
Upstream Pon Invalid Packets                              0
Upstream Dropped Packets                                  0
Upstream Dropped Ploams Fifo Full                         0
Downstream TM Valid Packets                               0
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            0
Downstream TM Packets Dropped Gem Pid Not Enabled         0
Downstream TM Q0 Valid Packets                            0
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                              0
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
******  intf  ******
Interface Name                                            1-4-4-0
Operational Status                                        Down
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                                     *** n/a ***
Speed Megabits per Second                                 2400
```

3 When specifying **olt** as the stats type, only the GPON OLT physical layer statistics are displayed for this OLT interface.

```
zSH> port stats 1-1-3-0/gponolt olt
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
```

Table 44 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-194/198 to ONU, and the upstream stats was sent from ONU to MXK-194/198.

Table 44: 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.
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.

Table 44: GPON OLT physical layer statistics attributes

Attribute	Description
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.
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.

Table 44: GPON OLT physical layer statistics attributes

Attribute	Description
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-194/198. 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.

View ONU statistics

Viewing ONU stats

1 View the ONU interfaces. This step is optional.

```
zSH> list if-translate

Processing list of 874

if-translate  1-1-1-0/rs232
if-translate  1-1-1-0/eth
if-translate  1-1-1-0-eth/linegroup
if-translate  1-1-2-0/eth
if-translate  1-1-2-0-eth/linegroup
if-translate  1-1-3-0/eth
if-translate  1-1-3-0-eth/linegroup
if-translate  1-1-4-0/eth
if-translate  1-1-4-0-eth/linegroup
if-translate  1-1-5-0/eth
if-translate  1-1-5-0-eth/linegroup
if-translate  1-1-6-0/eth
if-translate  1-1-6-0-eth/linegroup
if-translate  1-1-7-0/eth
if-translate  1-1-7-0-eth/linegroup
if-translate  1-1-8-0/eth
if-translate  1-1-8-0-eth/linegroup
if-translate  1-1-9-0/eth
if-translate  1-1-9-0-eth/linegroup
if-translate  1-1-6-0/ipobridge
if-translate  ipobridge/linegroup
if-translate  1-1-1-0/gponolt
if-translate  1-1-1-0-gponolt/linegroup
if-translate  1-1-2-0/gponolt
if-translate  1-1-2-0-gponolt/linegroup
if-translate  1-1-3-0/gponolt
if-translate  1-1-3-0-gponolt/linegroup
if-translate  1-1-4-0/gponolt
if-translate  1-1-5-0/gponolt
if-translate  1-1-5-0-gponolt/linegroup
if-translate  1-1-6-0/gponolt
if-translate  1-1-7-0/gponolt
if-translate  1-1-7-0-gponolt/linegroup
if-translate  1-1-8-0/gponolt
if-translate  1-1-8-0-gponolt/linegroup
if-translate  1-1-1-1/gpononu
if-translate  1-1-1-2/gpononu
...
874 entries found.
```

2 When specifying **onu** as the stats type, the ONU physical layer statistics are displayed for this ONU interface.

```
zSH> port stats 1-1-1-1/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 specifying **all** as the stats type, only ONU stats type is available and displayed for this ONU interface.

```
zSH> port stats 1-1-1-1/gpononu all
******  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
```

Table 45 defines the GPON ONU physical layer statistics displayed in the **port stats** *ifName*/**gpononu** command.

Table 45: 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.

Table 45: GPON ONU physical layer statistics attributes

Attribute	Description
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.
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 sent from the ONU.

System maintenance

This section describes the following:

- Change the serial craft port settings, page 566
- Manually binding interfaces, page 568
- Rename interfaces, page 567
- Activate or deactivate interfaces, page 568
- Save and restore configurations, page 570
- SNTP, page 571
- User accounts, page 571
- SFP presence and status, page 575
- View chassis information, page 578
- View and set MXK-194/198 time and day, page 579

Change the serial craft port settings

Tip: You only need to modify an **rs232-profile** if you want to change the default configuration of the serial craft port.

The MXK-194/198 **rs232-profile** can be used to configure serial craft ports on the system.

The default settings for the MXK-194/198 serial control ports are:

- 9600bps
- 8 data bits

- No parity
- 1 stop bit
- No flow control

Changing the serial control port settings

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

Update the **rs232-profile**. For example:

```
zSH> update rs232-profile 1-1-1-0/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 new record? [s]ave, [c]hange or [q]uit: s
Record created.
```

The settings take effect after the profile is saved.

Note: If the **rs232-profile** is deleted, the port speed is set to the last configured value.

Rename interfaces

The MXK-194/198 supports customized interface names using up to 32 characters. The customized name appears in bulk statistics and other output displaying interface names.

To customize an interface name, update the **ifName** parameter in the if-translate profile for the interface.

```
zSH> update if-translate 1-1-1-0/gponolt
if-translate  1-1-1-0/gponolt
Please provide the following: [q]uit.
ifIndex: ----------->  {23}:
shelf: ------------->  {1}:
slot: -------------->  {1}:
port: -------------->  {1}:
subport: ----------->  {0}:
type: -------------->  {other}:
adminstatus: ------->  {up}:
```

```
physical-flag: ----->  {true}:
iftype-extension: -->  {gponolt}:
ifName: ------------>  {1-1-1-0}:[interfacename upto 32 characters]
redundancy-param1: ->  {0}:
description-index: ->  (0):  **ready-only**
....................
Save changes? [s]ave, [c]hange or [q]uit: s
Record updated.
```

Activate or deactivate interfaces

Physical interfaces on the MXK-194/198 have associated **if-translate** profiles, which enable or disable the interfaces. To change the admin state of an interface, you can use the **adminstatus** parameter in the **if-translate** profile associated with the interface, or use the **port up | down | bounce** *ifIndex/type* command.

For example, to deactivate a MXK-194/198 GPON interface, you can use this **if-translate** profile:

```
zSH> update if-translate 1-1-1-0/gponolt
if-translate  1-1-1-0/gponolt
Please provide the following: [q]uit.
ifIndex: ----------->  {23}:
shelf: ------------->  {1}:
slot: -------------->  {1}:
port: -------------->  {1}:
subport: ----------->  {0}:
type: -------------->  {other}:
adminstatus: ------->  {up}:down
physical-flag: ----->  {true}:
iftype-extension: -->  {gponolt}:
ifName: ------------>  {1-1-1-0}:
redundancy-param1: ->  {0}:
description-index: ->  {0}:  ** read-only **
....................
Save changes? [s]ave, [c]hange or [q]uit:s
```

or use this **port down** command:

```
zSH> port down 1-1-1-0/gponolt
```

Manually binding interfaces

When creating **ip-interface-record** profiles, the syntax is *name*/*type*. The name of the IP interface can be user-defined or match the naming of the **if-translate** record for the physical interface. The system automatically binds interfaces if the name of the new IP record matches the name of the **if-translate** profile or if the syntax *1*/*1*/*port*/*subport*/*type* is used. Enter a **list if-translate** command to determine what **if-translate** records are available on your system.

The example below shows a new **ip-interface-record** being created with a user-defined name.

```
zSH> new ip-interface-record myip/ip
Please provide the following: [q]uit.
vpi: --------------->  {0}:
vci: --------------->  {0}:
rdindex: ----------->  {1}:
dhcp: -------------->  {none}: ** read-only **
addr: -------------->  {0.0.0.0}: 192.168.88.200
netmask: ----------->  {0.0.0.0}: 255.255.255.0
bcastaddr: --------->  {0.0.0.0}: 192.168.88.255
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}:
ingressfilterid: --->  {0}:
egressfilterid: ---->  {0}:
ipaddrdynamic: ----->  {static}:
dhcpserverenable: -->  {false}:
subnetgroup:-------->  {47}:
unnumberedindex:---->  {47}:
mcastcontrollist:--->  {264}:
vlanid:------------->  {95}:
maxVideoStreams:---->  {6}:
tosOption: -------------------->  {disable}:
tosCOS: ---------------------->  {0}:
vlanCOS: --------------------->  {0}:
s-tagTPID: ------------------->  {0x8100}:
s-tagId: --------------------->  {0}:
s-tagIdCOS: ------------------->  {0}:
....................
Save new record? [s]ave, [c]hange or [q]uit: s
Cannot determine binding for this IP interface.
Could not automatically bind this IP Interface.
New record saved.
```

Since the system did not automatically bind the new IP interface, manually bind the interface with the **stack bind** command:

```
zSH> stack bind
Enter the upper layer: myip/ip   the IP interface created
Enter the lower layer: 1-1-1-0-eth/other   the line group associated with Ethernet
Stack bind successful.
```

Note: The **stack bind** command does not allow binding directly to physical interfaces. You must bind two logical interfaces.

Enter the **stack show** command (with *name*/*type* syntax) to see interface binding:

```
zSH> stack show myip/ip
Line Group: 1-1-1-0-eth/other
Physical: 1/1/1/0/eth
```

Save and restore configurations

The **dump** and **restore** commands enable you to save and restore the system configuration. You can save the configuration to the console, a local file, or the network.

The command uses the following syntax:

`dump [console] [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** and **restore** commands use 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.

Note: The **dump** and **restore** commands use Secure FTP(SFTP) when the **system 0** profile secure mode is enabled.

To save the configuration to a console:

1. Configure your terminal emulation software as follows:
   - 9600bps
   - 8 data bits
   - No parity
   - 1 stop bit
   - No hardware flow control
   - VT100
   - Set Line Delay and Character Delay to 40 milliseconds
2. Turn on the file capture utility of your terminal emulation software.
3. Save the configuration by entering:

   `dump console`

   Do not press the Enter key.

4 Start the capture utility on your terminal emulation software and enter a name for the file (use a **.tx**t extension).

5 Press the Enter key.

The configuration file will be displayed on the screen.

6 When configuration file is finished, stop the capture utility.

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 dvice.cfg on the host 192.168.8.21:

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

Restoring the configuration

For information on restoring your configuration, refer to the release notes for your release.

SNTP

To set up the system to use SNTP:

Update the **ntp-client-config** profile. For example:

```
zSH> update 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.
```

User accounts

MXK-194/198 users have access to the CLI and are able to configure and administer the system.

user command

The **user** command enables the command line feature to add, modify, show, or delete users and user settings.

Syntax

user add <user-name> [password password] [prompt prompt] [admin] [zhonedebug] [voice] [data] [manuf] [dbase] [systems] [tools] [useradmin] [all]

user modify <user-name> [password *password*] [prompt *prompt*] [admin] [zhonedebug] [voice] [data] [manuf] [dbase] [systems] [tools] [useradmin] [all]

user delete <user-name>

user show [<user-name>]

Options

add
Adds a new user profile with the specified settings.

username
Name of the user.

password password
Specifies the password assigned to this user.

prompt
Specifies the system prompt to display for this user. If no password is entered, the system assigns a random password. Enclosing an argument in quotes allows the entry of special characters.

access level
Specifies the access levels assigned to the user. The **all** option sets all access levels. Individual access levels can be specified by added the keyword true or false after an access level. For example, **manuf false all true** sets all access levels except manuf level access.

Example 1

```
zSH> user add steve password pass prompt "zSH >" admin voice systems dbase
User record saved.
.................................
User name:(Steve)  User prompt:(zSH >)
Access Levels:
(admin)(voice)(system)(dbase)
```

Example 2

```
zSH> user modify joe password pass all false admin true
OK to modify this account? [yes] or [no]: yes
User record updated.
.................................
User name:(newaccount2)  User prompt:(zSH>)
Access Levels:
(admin)(useradmin)
```

Example 3

```
zSH> user show
..................................
User name:(admin)  User prompt:(zSH>)
Access Levels:
(admin)(voice)(data)(manuf)(database)(systems)(tool)(useradmin)
..................................
User name:(steve)  User prompt:(zSH>)
Access Levels:
(admin)(voice)(systems)(dbase)
..................................
User name:(joe)  User prompt:(test >)
Access Levels:
(admin)
..................................
User name:(kathy)  User prompt:(test4 >)
Access Levels:
(admin)(zhonedebug)(voice)(data)(manuf)(database)(systems)(tool)(useradmin)

zSH> user show steve
..................................
User name:(steve)  User prompt:(zSH>)
Access Levels:
(admin)(voice)(systems)(dbase)
```

Example 4

```
zSH> user delete kathy
OK to delete this account? [yes] or [no]: yes
Account kathy deleted
```

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. The highest level of access is **useradmin**, which allows the creation of user accounts.

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
```

Commands with **zhonedebug** privilege levels are intended for use by Zhone development only.

Immediately after activating the user account, you should change the password something you can remember, as explained in the next section.

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 jsmith
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 jsmith
Password:
```

SFP presence and status

If you need to verify the status of an SFP on an MXK-194/198 Ethernet port or a GPON port, use the **sfp show** command. This command also displays parameters of existing SFPs for diagnostics.

To check for ethernet interfaces on the MXK-194/198, enter **list ether**:

```
zSH> list ether
ether  1-1-1-0/eth
ether  1-1-2-0/eth
```

```
ether  1-1-3-0/eth
ether  1-1-4-0/eth
ether  1-1-5-0/eth
ether  1-1-6-0/eth
ether  1-1-7-0/eth
ether  1-1-8-0/eth
ether  1-1-9-0/eth
9 entries found.
```

To check for GPON interfaces on the MXK-194/198, enter **list gpon-olt-config**:

```
zSH> list gpon-olt-config
gpon-olt-config  1-1-1-0/gponolt
gpon-olt-config  1-1-2-0/gponolt
gpon-olt-config  1-1-3-0/gponolt
gpon-olt-config  1-1-4-0/gponolt
gpon-olt-config  1-1-5-0/gponolt
gpon-olt-config  1-1-6-0/gponolt
gpon-olt-config  1-1-7-0/gponolt
gpon-olt-config  1-1-8-0/gponolt
8 entries found.
```

To view SFP parameters on an particular interface, enter **sfp show** *interface/type*:

```
zSH> sfp show 1-1-1-0/gponolt
vendorName                                FUJITSU
vendorOui                                 00-00-0e
vendorPartNumber                          FIM30538
vendorRevisionLevel                       03A
serialNumber                              02329  E36476
manufacturingDateCode                     091223
complianceCode                            unknown value (0x0000)
connectorType                             sc (1)
transceiverType                           sfp (3)
extendedIdentifier                        4
encodingAlgorithm                         nrz (3)
channelLinkLength                         unknown value (0x0000)
channelTransmitterTechnology              unknown value (0x0000)
channelTransmitterMedia                   unknown value (0x0000)
channelSpeed                              unknown value (0x0000)
nineTo125mmFiberLinkLengthKm              20
nineTo125mmFiberLinkLength100m            200
fiftyTo125mmFiberLinkLength10m            0
sixtyTwoDot5To125mmFiberLinkLength10m     0
nominalBitRate                            0
upperBitRateMarginPercentage              0
lowerBitRateMarginPercentage              0
copperLinkLength                          0
```

To see if any SFPs are present on a MXK-194/198, enter the **sfp show all**:

```
zSH> sfp show all
```

```
SFP Data for interface 1-1-1-0/gponolt
vendorName                                FUJITSU
vendorOui                                 00-00-0e
vendorPartNumber                          FIM30538
vendorRevisionLevel                       03A
serialNumber                              02329  E36476
manufacturingDateCode                     091223
complianceCode                            unknown value (0x0000)
connectorType                             sc (1)
transceiverType                           sfp (3)
extendedIdentifier                        4
encodingAlgorithm                         nrz (3)
channelLinkLength                         unknown value (0x0000)
channelTransmitterTechnology              unknown value (0x0000)
channelTransmitterMedia                   unknown value (0x0000)
channelSpeed                              unknown value (0x0000)
nineTo125mmFiberLinkLengthKm              20
nineTo125mmFiberLinkLength100m            200
fiftyTo125mmFiberLinkLength10m            0
sixtyTwoDot5To125mmFiberLinkLength10m     0
nominalBitRate                            0
upperBitRateMarginPercentage              0
lowerBitRateMarginPercentage              0
copperLinkLength                          0

SFP Data for interface 1-1-2-0/eth
** No SFP present **

SFP Data for interface 1-1-2-0/gponolt
vendorName                                FUJITSU
vendorOui                                 00-00-0e
vendorPartNumber                          FIM30538
vendorRevisionLevel                       03A
serialNumber                              02239  E36512
manufacturingDateCode                     091222
complianceCode                            unknown value (0x0000)
connectorType                             sc (1)
transceiverType                           sfp (3)
extendedIdentifier                        4
encodingAlgorithm                         nrz (3)
channelLinkLength                         unknown value (0x0000)
channelTransmitterTechnology              unknown value (0x0000)
channelTransmitterMedia                   unknown value (0x0000)
channelSpeed                              unknown value (0x0000)
nineTo125mmFiberLinkLengthKm              20
nineTo125mmFiberLinkLength100m            200
fiftyTo125mmFiberLinkLength10m            0
sixtyTwoDot5To125mmFiberLinkLength10m     0
nominalBitRate                            0
upperBitRateMarginPercentage              0
lowerBitRateMarginPercentage              0
copperLinkLength                          0

SFP Data for interface 1-1-3-0/eth
```

```
** No SFP present **

<SPACE> for next page, <CR> for next line, A for all, Q to quitq
```

View chassis information

The following commands display information about the status of the system:

- **shelfctrl**
- **slots**

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

```
zSH> shelfctrl monitor
Shelf                           Status
-------------------------------------------------------------------------
Uptime                          7 days, 5 hours, 2 minutes
Temperature Sensor              Celsius(C)              Fahrenheit(F)
-------------------------------------------------------------------------
Card sensor                     30                      86
Temperature reading             normal
Fans                            Status
-------------------------------------------------------------------------
Fan A                           normal
Fan B                           normal
Fan C                           normal
System Alarm                    Status
-------------------------------------------------------------------------
System                          Critical alarm set
```

To view general system statistics:

```
zSH> shelfctrl stats
Shelf Controller Message Statistics
-----------------------------------
Directory services: 2
Clock: 5049
Receive errors: 2
```

To verify whether the shelf is active:

```
zSH> shelfctrl show
Shelf Controller Address: 01:1:12
Shelf Registry Address: 01:1:1045
Lease ID: 0x02070000_00000033
State: active
Slot 1:
   prevState: CONFIGURING currentState: RUNNING
   mode: NONE startTime: 664425249
```

To view information about the device, use the **slots** command:

```
zSH> slots 1
Type            : MXK 19x
```

```
Sub-Type        : MXK 198 - 8 GPON OLT, 8 FE/GE
Card Version    : 00001
EEPROM Version  : 2
Serial #        : 2361518
CLEI Code       : No CLEI
Card-Profile ID : 1/1/10500
Shelf           : 1
Slot            : 1
ROM Version     : MXK 2.2.1.211
Software Version: MXK 2.2.1.226
State           : RUNNING
Mode            : NONE
Heartbeat check : enabled
Heartbeat resp  : 0
Heartbeat late  : 0
Hbeat seq error : 0
Hbeat longest   : 0
Fault reset     : enabled
Power fault mon : not supported
Uptime          : 7 days, 5 hours, 4 minutes
```

View and set MXK-194/198 time and day

Viewing the MXK-194/198 time and day

View the current time on the MXK-194/198, if necessary, with **showdatetime**:

```
zSH> showdatetime
Current Time: TUE FEB 09 09:49:52 2010
```

Setting the MXK time and day

Set the MXK-194/198 time and day if not already set.

You can either use the **setdatetime** command or use SNTP (the network time protocol) to provide time to the MXK-194/198 rather than set it manually. For example:

```
zSH> setdatetime < month(mm) day(dd) year(yyyy) hour(hh) minute(mm) second(ss) >
```

INDEX

Numerics

802.1 Q-in-Q (VLAN tagging) 286
802.1p priority queues 154

A

acronyms, described 15
adding a user, description of 571
admin
    deleting user account of 574
administration
    configuring traps 537
    creating SNMP access lists 535
    creating SNMP community names 535
    logging 525
    saving and restoring configurations 570
    user accounts 571
airflow, system requirements for 37
alarm terminals 113
alarms 116
    viewing 578

B

battery, safety precautions for 34
binding interfaces 568
bridged video 307
bridging
    intralinks 225
bridging commands 304
bridging configuration
    bridged video 307
    bridging configurations 281
    downlink bridge tagged on Active Ethernet 284
    downlink bridge untagged on Active Ethernet 283
    downlink bridges tagged or untagged with VLAN ID 283
    link aggregation 298
    Q-in-Q 286
    Q-in-Q downlink and uplink stagged 290
    tagged to stagged configuration 287
    tagged uplink bridge with VLAN ID 282
    TLS bridging 293

C

cable specifications 44
cables
    cleaning fiber optic 48
    optical 122
cables and connectors
    cabling guidelines 43
        NEC article 800 43
        power lines 43
    ground minimum 110
    ratings 39
    rules 39
    using ducts 37
card-profile, setting 55
change default passwords, how to 574
chassis
    dedicated ground 35
    dimensions 38
    environmental specifications 38
    fans 38
    maximum temperature 37
    number per rack 38
    operating altitude 38
    operating humidity 38
    operating temperature 38
    rack installation 107
    storage altitude 38
    storage humidity 38
    storage temperature 38
    unpacking 105
    viewing errors 578
    viewing temperature 578
    weight 38
    weight distribution 37
chassis dimensions 38
Class of Service (COS) 239
Class of Service (CoS) 153
cleaning components 48
clock
    setting system using NTP 571
commands
    log 533
    log show 533
    rip 152
    stack bind 569
common return 39
compliance, specifications supported 41
configuration
    DNS resolver 143
    logging in 56
    logging out 56
    overview of profiles 54

RIP 152
saving and restoring 570
configuring a management interface 57
configuring IP
DNS resolver 143
RIP 152
configuring traps, description of 537
connecting power
power supplies 111
procedure 111
terminal block 111
CoS 153
CoS parameters 154
COS, in VLAN headers 239
CPE manager 70
CPE profile
copy
CPE profile
move 413
creates 183
creating SNMP access lists, description of 535
creating SNMP community names, description of 535
cutoff requirements 39

D

default configuration, description of 54
default passwords, changing 574
deleting a user, description of 574
destination MAC swapping 265
device settings 55
DHCP
configurations 146
DHCP relay 150
DHCP server options 147
DHCP server support 146
overview 145
server profiles 146
DHCP configurations 146
DHCP logging 196
DHCP relay 150, 260
DHCP relay agent 179, 260
DHCP server 179
DHCP server profiles 146
DNS resolver configuration
creating a host profile 143
creating a resolver record 143
DNS, description of 141
Domain Name System, see DNS
domain parameter 143
downlink bridge tagged on Active Ethernet 284
downlink bridge untagged on Active Ethernet 283
downlink bridges tagged or untagged with VLAN ID 283
Dynamci OMCI
Dynamic OMCI overview 363
dynamic IP filtering on a bridge
secure DHCP 300

E

environmental requirements
precautions 38
specifications 38
errors, viewing system 578
Ethernet interface
configuration 57
creating a default route 58
IP interface 57
route show command 58
verifying 58
verifying the route 58

F

fan tray

Hot swappable 117
FE/GE uplink specifications 42
fiber
cleaning cables 48
first-nameserver parameter 143
floodMulticast 295
floodUnknown 295
Forbid OUI 260

G

Generic profile
creation 340
definition 332
deletion 353
import/export 357, 413
GPON card
CPE profile 413
Dynamic OMCI overview 363
extended reach 447

Generic profile 332, 340, 353, 357, 413
ME profile 332, 340, 353, 357
OMCI overview 331
Smart OMCI configuration 335
Smart OMCI overview 331
Specific profile 332, 353, 357, 413
grounding
acceptable conductors 35
cable gauge 110
conductor requirements 108
dedicated 35
power 35
specifications for 108
system 108
using power supply connection 37

H

host profile 144
hostalias1 parameter 144
hostalias2 parameter 144
hostalias3 parameter 144
hostalias4 parameter 144
host-based routing 137
host-based routing configurations 167
host-based routing with an external DHCP server 179
host-based routing with external DHCP server 179
host-based routing with external DHCP server and alternate DHCP server with dhcp-relay agent 187
host-based routing with external DHCP server for DNS and bootp services 175
host-based routing with external DHCP server to provide DNS and bootp services 175
host-based routing with MXK as local DHCP server 171
host-based routing without DHCP 168
hostname parameter 144
HTTPS (HTTP secure) 89

I

IGMP snooping with proxy reporting 313
installation

unpacking the system 105
connecting power 111
grounding conductors 108
rack installation 107
installation precautions 37
airflow 37
cables and connectors 37
cabling ducts 37
chassis weight 37
environmental requirements 38
grounding 37
maximum temperature 37
ventilation 37
weight distribution 37
intermediate agent, PPPoE 246
Intralinks
configuring 225
IP 131
overview 141
QoS 153
IP address
assigning to Ethernet 57
IP administrative procedures 192
IP interface
Class of Service (CoS) 153
Type of Service (ToS) 153
IP routing
host-based routing 137
IP statistics commands 198
IPSLA 142

L

laser beam, safety precautions for 34
LEDs
description 119
system described 119
link aggregation 298
log messages, description of content for 525
logging
description 525
displaying persistent logs 535
enabling/disabling 528
enabling/disabling for session 56
enabling/disabling over the serial craft port 56
log messages 525
modifying logging levels 533
syslog, configuring 528
logging levels, log command and modifying 533

M

maintenance
cleaning toolkit 49
management
Zhone Web Config Tool 103
management interfaces 41
maximum temperature, precautions and 37
ME profile
creation 340
definition 332
deletion 353
import 353
import/export 357
MTAC/Ring card
external alarm contacts 116
MTAC/Ring external contacts 116
multicast
creating control list 310
MXK-194/198 configuration
CPE manager 70
Web UI cut-through for Fast Ethernet ONT devices 81

N

network-based routing with an external DHCP server 164
network-based routing with the MXK as local DHCP server 160
network-based routing without DHCP 157
NTP
configuring 571

O

OMCI overview 331

P

packet-rule-record
destination MAC swapping 265
DHCP relay 260
Forbid OUI 260
parameters
domain 143
first-nameserver 143
hostalias1 144
hostalias2 144
hostalias3 144
hostalias4 144
hostname 144
query-order 143
second-nameserver 143
third-nameserver 143
passwords, changing default 574
persistent logs, displaying 535
power
using supply for grounding 37
power specifications
cable ratings 39
cables and connectors 39
common return 39
cutoff requirements 39
PPPoE intermediate agent 246
preparing for installation
installation precautions 37
safety precautions 33
selecting the system location 37
tools you need 36
profiles
host 144
overview of configuration 54
resolver 143, 144

Q

Q-in-Q 286
Q-in-Q downlink and uplink stagged 290
Q-in-Q parameters 286
Q-in-Q stagged 287
Q-in-Q tagged 287
QoS 153
query-order parameter 143

R

rack installation
chassis 107
procedure 107
rear alarm terminals 113
resetting passwords, description of 575
resolver profile 143, 144
RIP
configuration 152
configuring global defaults 152
description 141
rip command 152

Routing Information Protocol, see RIP

S

safety
    standards 33
safety precautions
    battery 34
    description 34
    laser beam 34
Saving and restoring configurations 571
saving and restoring configurations
    description 566
second-nameserver parameter 143
secure DHCP 300
secure shell (SSH) 89
security
    HTTPS (HTTP secure) 89
    Secure File Transfer Protocol (SFTP) 89
    secure shell (SSH) 89
selecting the system location 37
service level agreement, SLA 142
SFPs
    supported SFPs 31
Smart OMCI
    Smart OMCI overview 331
    Smart OMCI web-interface 335
SNMP
    statistics 537
Specific profile
    definition 332
    deletion 353
    import/export 357, 413
specifications
    chassis dimensions 38
    environmental 38
stack bind 569
stagged bridge 286
statistics, SNMP 537
symmetrical bridging 293
syslog server, configuring 528
system
    default configuration 54
    environmental dimensions
        chassis 38
        weight 38
    Ethernet interface 57
    maximum temperature 37
    profile
        updating 60
    system profile 60
    unpacking 105
    uplinks 57
system configuration
    DHCP configurations 146
system environmental dimensions
    chassis per rack 38
    fans 38
    operating altitude 38
    operating humidity 38
    operating temperature 38
    storage altitude 38
    storage humidity 38
    storage temperature 38
system specifications 41

T

tagged to stagged bridge configuration 287
temperature, maximum 37
temperature, viewing chassis 578
TFTP server 360
third-nameserver parameter 143
TLS bridge
    floodUnknown and floodMulticast 295
tools for installation 36
ToS 153
ToS parameters 154
traps
    configuring 537
Type of Service (ToS) 153

U

uplink bridge tagged with VLAN ID 282
uplink specifications 42
uplinks
    configuration 57
user accounts 571
    adding a user 571
    changing default passwords 574
    deleting a user 574
    deleting admin 574
    resetting passwords 575

V

ventilation, requirements for 37

video
    multicast control list 310
video bridged 307
VLAN
    Fields in the VLAN header 154
VLAN header 154
VLAN ID and SLAN ID 286

W

Web Configuration Tool
    configuration, Web tool 103
Web UI cut-through for Fast Ethernet ONT devices
    81