Configuring Link Aggregation on the ML-Series Card

This chapter describes how to configure link aggregation for the ML-Series cards, both EtherChannel and packet-over-SONET (POS) channel. For additional information about the Cisco IOS commands used in this chapter, refer to the *Cisco IOS Command Reference* publication.

This chapter contains the following major sections:

Understanding Link Aggregation

The ML-Series card offers both EtherChannel and POS channel. Traditionally EtherChannel is a trunking technology that groups together multiple full-duplex IEEE 802.3 Ethernet interfaces to provide fault-tolerant high-speed links between switches, routers, and servers. EtherChannel forms a single higher bandwidth routing or bridging endpoint and was designed primarily for host-to-switch connectivity. The ML-Series card extends this link aggregation technology to bridged POS interfaces. POS channel is only supported with LEX encapsulation.

Link aggregation provides the following benefits:

- Logical aggregation of bandwidth
- Load balancing
- Fault tolerance

Port channel is a term for both POS channel and EtherChannel. The port channel interface is treated as a single logical interface although it consists of multiple interfaces. Each port channel interfaces consists of one type of interface, either Fast Ethernet or POS. You must perform all port channel configurations on the port channel (EtherChannel or POS channel) interface rather than on the individual member Ethernet or POS interfaces. You can create the port channel interface by entering the **interface port-channel** interface configuration command.

Port channel connections are fully compatible with IEEE 802.1Q trunking and routing technologies. IEEE 802.1Q trunking can carry multiple VLANs across a port channel.

Each ML100-FX supports up to four FECs plus an additional POS channel, a port channel made up of the two POS ports. A maximum of four Fast Ethernet ports can bundle into one Fast Ethernet Channel (FEC) and provide bandwidth scalability up to 400-Mbps full-duplex Fast Ethernet.

Caution The EtherChannel interface is the Layer 2/Layer 3 interface. Do not enable Layer 3 addresses on the physical interfaces. Do not assign bridge groups on the physical interfaces because doing so creates loops.

Caution Before a physical interface is removed from an EtherChannel (port channel) interface, the physical interface must be disabled. To disable a physical interface, use the **shutdown** command in interface configuration mode.

Note Link aggregation across multiple ML-Series cards is not supported.

Note Policing is not supported on port channel interfaces.

Note The ML-Series does not support the routing of Subnetwork Access Protocol (SNAP) or Inter-Switch Link (ISL) encapsulated frames.

Configuring Link Aggregation

You can configure an FEC or POS channel by creating an EtherChannel interface (port channel) and optionally assigning a network IP address.

Configuring Fast EtherChannel

All interfaces that are members of an FEC should have the same link parameters, such as duplex and speed.

To create an EtherChannel interface, perform the following procedure, beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface port-channel channel-number`	Creates the EtherChannel interface.
Step 2	`Router(config-if)# ip address ip-address subnet-mask`	(Optional) Assigns an IP address and subnet mask to the EtherChannel interface.
Step 3	`Router(config-if)# end`	Exits to privileged EXEC mode.
Step 4	`Router# copy running-config startup-config`	(Optional) Saves configuration changes to NVRAM.

For information on other configuration tasks for the EtherChannel, refer to the *Cisco IOS Configuration Fundamentals Configuration Guide*.

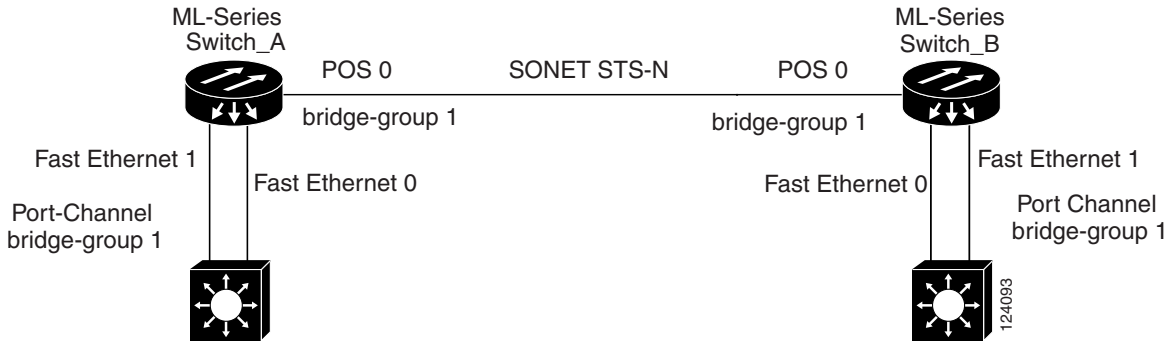
To assign Ethernet interfaces to the EtherChannel, perform the following procedure, beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface fastethernet number`	Enters one of the interface configuration modes to configure the Fast Ethernet interface that you want to assign to the EtherChannel.
Step 2	`Router(config-if)# channel-group channel-number`	Assigns the Fast Ethernet interface to the EtherChannel. The channel number must be the same channel number you assigned to the EtherChannel interface.
Step 3	`Router(config-if)# end`	Exits to privileged EXEC mode.
Step 4	`Router# copy running-config startup-config`	(Optional) Saves configuration changes to NVRAM.

EtherChannel Configuration Example

Figure 10-1 shows an example of encapsulation over EtherChannel. The associated commands are provided in Example 10-1 and Example 10-2.

Figure 10-1 Encapsulation over EtherChannel Example



Example 10-1 ML_Series A Configuration

```
hostname Switch A
no ip routing
!
bridge 1 protocol ieee
!
interface Port-channel 1
bridge-group 1
 hold-queue 150 in
!
interface FastEthernet 0
channel-group 1
!
```

```
interface FastEthernet 1
channel-group 1
!
interface POS 0
bridge-group 1
```

Example 10-2 ML_Series B Configuration

```
hostname Switch B
no ip routing
!
bridge 1 protocol ieee
!
interface Port-channel 1
bridge-group 1
 hold-queue 150 in
!
interface FastEthernet 0
channel-group 1
!
interface FastEthernet 1
channel-group 1
!
interface POS 0
bridge-group 1
!
```

Configuring POS Channel

You can configure a POS channel by creating a POS channel interface (port channel) and optionally assigning an IP address. All POS interfaces that are members of a POS channel should have the same port properties and be on the same ML-Series card.

Note POS channel is only supported with G-Series card compatible (LEX) encapsulation.

To create a POS channel interface, perform the following procedure, beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface port-channel` *channel-number*	Creates the POS channel interface. You can configure one POS channel on the ML-Series card.
Step 2	`Router(config-if)# ip address` *ip-address subnet-mask*	Assigns an IP address and subnet mask to the POS channel interface (required only for the Layer 3 POS channel).
Step 3	`Router(config-if)# end`	Exits to privileged EXEC mode.
Step 4	`Router# copy running-config startup-config`	(Optional) Saves configuration changes to NVRAM.

Caution The POS channel interface is the routed interface. Do not enable Layer 3 addresses on any physical interfaces. Do not assign bridge groups on any physical interfaces because doing so creates loops.

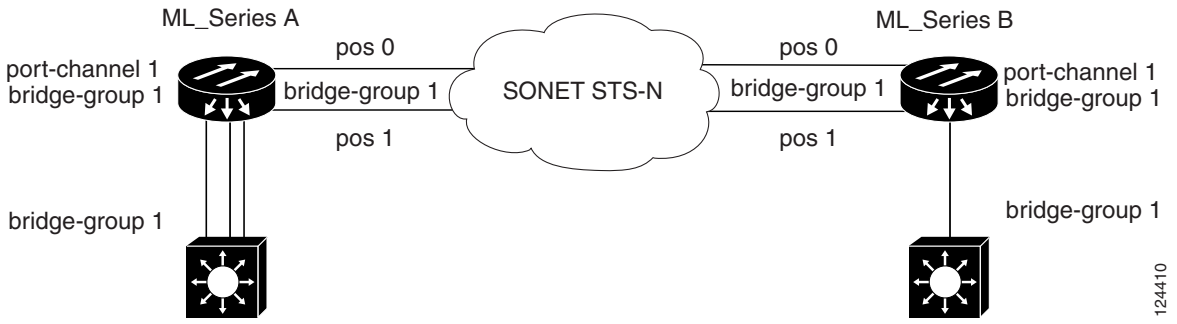
To assign POS interfaces to the POS channel, perform the following procedure, beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface pos` *number*	Enters the interface configuration mode to configure the POS interface that you want to assign to the POS channel.
Step 2	`Router(config-if)# channel-group` *channel-number*	Assigns the POS interface to the POS channel. The channel number must be the same channel number that you assigned to the POS channel interface.
Step 3	`Router(config-if)# end`	Exits to privileged EXEC mode.
Step 4	`Router# copy running-config startup-config`	(Optional) Saves the configuration changes to NVRAM.

POS Channel Configuration Example

Figure 10-2 shows an example of POS channel configuration. The associated code for ML_Series A is provided in Example 10-3 and for ML_Series B in Example 10-4.

Figure 10-2 POS Channel Example



Example 10-3 ML_Series A Configuration

```
no ip routing
bridge 1 protocol ieee
!
!
interface Port-channel1
 no ip address
bridge-group 1
!
interface FastEthernet0
 no ip address
 bridge-group 1
```

```
!
interface POS0
channel-group 1
!
interface POS1
channel-group 1
```

Example 10-4 ML_Series B Configuration

```
bridge irb
bridge 1 protocol ieee
!
!
interface Port-channel1
bridge-group 1
!
interface FastEthernet0
bridge-group 1
!
interface POS0
channel-group 1
!
interface POS1
 no ip address
 channel-group 1
```

Understanding Encapsulation over FEC or POS Channel

When configuring encapsulation over FEC or POS, be sure to configure IEEE 802.1Q on the port-channel interface, not its member ports. However, certain attributes of port channel, such as duplex mode, need to be configured at the member port levels. Also make sure that you do not apply protocol-level configuration (such as an IP address or a bridge group assignment) to the member interfaces. All protocol-level configuration should be on the port channel or on its subinterface. You must configure IEEE 802.1Q encapsulation on the partner system of the EtherChannel as well.

Configuring Encapsulation over EtherChannel or POS Channel

To configure encapsulation over the FEC or POS channel, perform the following procedure, beginning in global configuration mode:

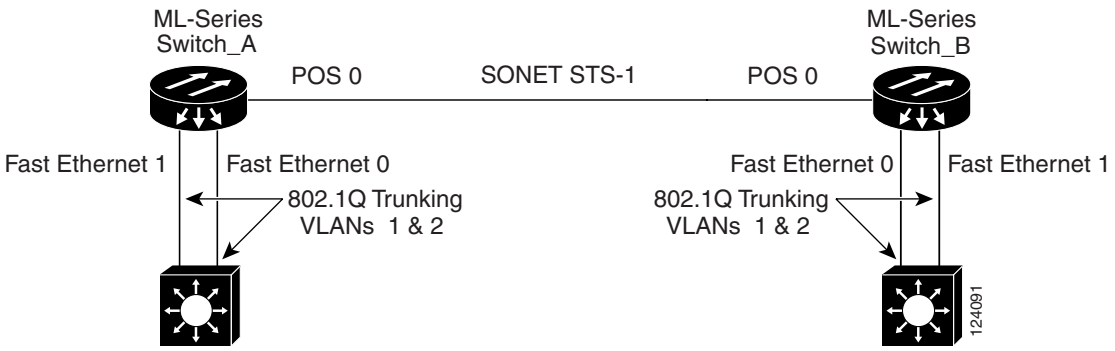
	Command	Purpose
Step 1	`Router(config)# interface port-channel` *channel-number.subinterface-number*	Configures the subinterface on the created port channel.
Step 2	`Router(config-subif)# encapsulation dot1q` *vlan-id*	Assigns the IEEE 802.1Q encapsulation to the subinterface.
Step 3	`Router(config-subif)# bridge-group` *bridge-group-number*	Assigns the subinterface to a bridge group.

	Command	Purpose
Step 4	`Router(config-subif)# end`	Exits to privileged EXEC mode. **Note** Optionally, you can remain in interface configuration mode and enable other supported interface commands to meet your requirements.
Step 5	`Router# copy running-config startup-config`	(Optional) Saves the configuration changes to NVRAM.

Encapsulation over EtherChannel Example

Figure 10-3 shows an example of encapsulation over EtherChannel. The associated code for ML_Series A is provided in Example 10-5 and for ML_Series B in Example 10-6.

Figure 10-3 Encapsulation over EtherChannel Example



This encapsulation over EtherChannel example shows how to set up two ONS 15310-CL nodes or ONS 15310-MA nodes with ML-Series cards to interoperate with two switches that also support IEEE 802.1Q encapsulation over EtherChannel. To set up this example, use the configurations in the following sections for both Switch A and Switch B.

Example 10-5 ML_Series A Configuration

```
hostname ML_Series_A
!
bridge irb
bridge 1 protocol ieee
bridge 2 protocol ieee
!
interface Port-channel1
hold-queue 150 in
!
interface Port-channel1.1
 encapsulation dot1Q 1 native
bridge-group 1
!
interface Port-channel1.2
 encapsulation dot1Q 2
bridge-group 2

!
```

```
interface FastEthernet0
channel-group 1
!
interface FastEthernet1
channel-group 1
!
interface POS0
!
interface POS0.1
 encapsulation dot1Q 1 native
 bridge-group 1
!
interface POS0.2
 encapsulation dot1Q 2
 bridge-group 2
```

Example 10-6 ML_Series B Configuration

```
hostname ML_Series_B
!
bridge irb
bridge 1 protocol ieee
bridge 2 protocol ieee
!
interface Port-channel1
hold-queue 150 in
!
interface Port-channel1.1
 encapsulation dot1Q 1 native
bridge-group 1
!
interface Port-channel1.2
 encapsulation dot1Q 2
bridge-group 2
!
interface FastEthernet0
channel-group 1
!
interface FastEthernet1
channel-group 1
!
interface POS0
!
interface POS0.1
 encapsulation dot1Q 1 native
 bridge-group 1
!
interface POS0.2
 encapsulation dot1Q 2
 bridge-group 2
!
```

Monitoring and Verifying EtherChannel and POS

After FEC or POS is configured, you can monitor its status using the **show interfaces port-channel** command.

Example 10-7 show interfaces port-channel Command

```
ML_Series# show int port-channel 9
Port-channel9 is down, line protocol is down
  Hardware is FEChannel, address is 0000.0000.0000 (bia 0000.0000.0000)
  Internet address is 192.26.24.22/25
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
    No. of active members in this channel: 0
  Last input never, output never, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/300/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/0 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes
     Received 0 broadcasts (0 IP multicast)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast
     0 input packets with dribble condition detected
     0 packets output, 0 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
```