AOS Configuration and Troubleshooting Guide

Frame Relay

This configuration guide will aid in the setup of Frame Relay for ADTRAN Operating System (AOS) products. An overview of Frame Relay general concepts combined with detailed command descriptions for example step-by-step assistance for configuration. The troubleshooting section outlines proper use of **show** and **debug** commands to verify that Frame Relay has been configured properly on the AOS product(s).

This guide consists of the following sections:

- *Frame Relay Overview*
- *Hardware/Software Requirements/Limitations*
- *Configuring Frame Relay in AOS*
    - *Configuring Basic Frame Relay Features in CLI*
    - *Configuring Additional Frame Relay Features in CLI*
    - *Configuring Frame Relay in the Web GUI*
- *Example Configurations*
- *Quick Configuration Guide*
- *Troubleshooting*

Frame Relay Overview

For companies that can accept lower transmission speeds during peak usage times, Frame Relay provides a more affordable WAN solution than a dedicated E1- or T1-carrier line. Frame Relay can run over a variety of physical WAN connections, including E1- and T1-carrier lines. Whatever the physical WAN connection is, Frame Relay allocates bandwidth on that connection dynamically. As a result, public carriers provide a subscriber with bandwidth only when that subscriber requires it.

Frame Relay cuts costs both for public carriers and subscribers because it minimizes idle bandwidth. Public carriers can allocate the same bandwidth to multiple subscribers; therefore subscribers do not pay for bandwidth that is not used.

When companies purchase Frame Relay service, Service Level Agreement (SLA) are negotiated specifying a Committed Information Rate (CIR), the amount of bandwidth that can be used. The CIR is contractually guaranteed bandwidth, rather than physically guaranteed as with dedicated E1- or T1- carrier lines. Failing to provide the CIR agreed upon in the SLA can result in a fine for the Frame Relay provider. Consequently, carriers usually ensure that the bandwidth stipulated in the CIR is available to the customer. (See Figure 1)

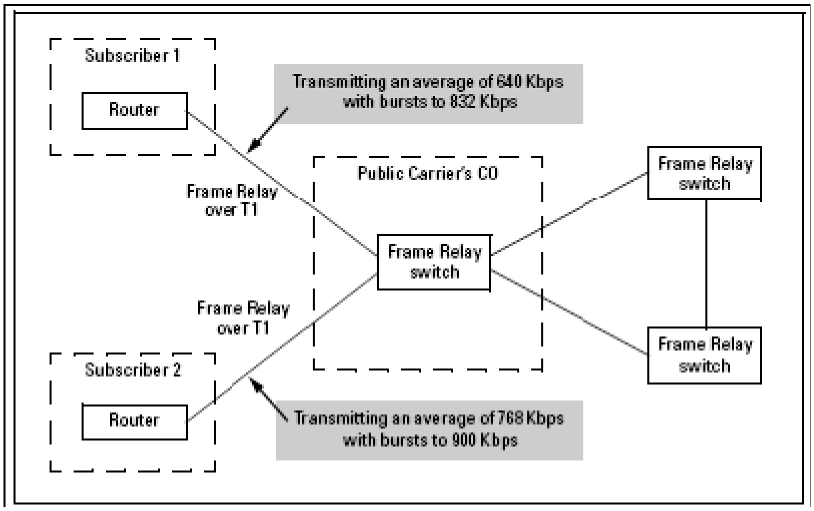


Figure 1: Frame Relay network dynamically allocates bandwidth.

Although a Frame Relay provider allows customers a set CIR, it is still possible to achieve a higher throughput than that amount for a limited time period. Frame Relay provides the ability to burst above the provided CIR in times of high traffic volume. All burst traffic can be dropped by the network if there is congestion at the provider’s switch. For this reason, all excess traffic is tagged as Discard Eligible (DE) and can be dropped if congestion occurs at any point in the provider’s network.

If for any reason this does occur, the Frame Relay Access Device (FRAD) at the user’s end will be notified of the drop by the sending of a Backward Explicit Congestion

Notification (BECN). A BECN is a notification that is sent towards the transmitting FRAD stating that it is sending too much information towards the Frame Relay Switch.

Packet Switched Network

Frame Relay transfers data through multiple nodes in a shared network using packet switching. Frame Relay divides data into frames, and each frame travels through the network individually, passing from one Frame Relay switch to another in a non-fixed path, until the frames are reassembled at their destination.

Although frames can take multiple and variable paths through a shared network, two routers, which are identified by administratively assigned circuit IDs, define the fixed endpoints to a permanent virtual circuit (PVC). In a Frame Relay network, a PVC is a logical connection between two sites.

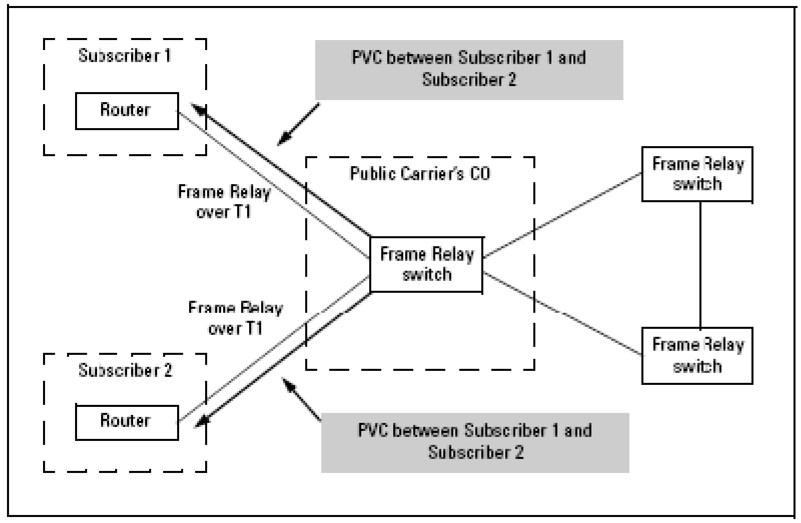


Figure 2 PVC connects two endpoints on the Frame Relay network.

Components of a Frame Relay Network

The Frame Relay network consists of several components, each of which has a specific role.

- User, or data terminal equipment (**DTE**)
- Network, or data communications equipment (**DCE**)
- Network-to-network interfaces (**NNI**)
- User-to-network interfaces (**UNI**)

When you configure Frame Relay on the AOS product, you must define the role that the router will perform in the Frame Relay network.

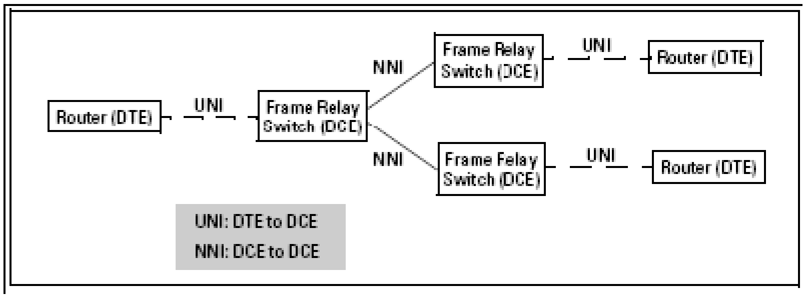


Figure 3. Components in a Frame Relay Network

DTE

The DTE receives data from the LAN in the form of multiple protocol packets and encapsulates each packet into a Frame Relay frame. In the header of such a frame is the Data Link Connection Identifier (DLCI), which contains the frame's ultimate destination. You can configure the DTE to manage congestion and maintain quality of service. For example, the DTE can manipulate the actual size of each frame sent through the network. It also can buffer and fragment packets to reserve bandwidth for particular circuits and ensure quality of service for time sensitive packets such as voice applications.

DCE

In a Frame Relay network, the DCE is the Frame Relay switch, which establishes and maintains the Frame Relay connection. After receiving frames from the DTE, the DCE converts these frames into signals supported by the physical media of the network. The DCE also reads the DLCI on incoming packets, checks its switch lookup table, and then forwards data to the appropriate outgoing port – which leads to the correct virtual endpoint.

UNI

UNIs connect the DTE to the DCE and provide access to the Frame Relay network.

NNI

NNIs connect a DCE to a DCE, using bidirectional signaling. That is, NNIs connect one Frame Relay switch to another.

DLCI

As mentioned earlier, the DTE marks each outgoing frame with a DLCI, a 10- bit field in the Address Field of the Frame Relay header. The switch reads the DLCI to determine the appropriate PVC endpoint to which to send the frame. DLCIs are locally, not globally, significant. (See Figure 4)

The 10-bit field enables 1024 possible DLCI numbers, but some are reserved for special purposes:

- 0 signals Annex A and D
- 1-15 and 1008-1022 are reserved
- 1023 signals Link Management Interface (LMI)

The remaining 976 DLCI numbers between 16 and 1007 are available to users. Your public carrier will assign you a DLCI.

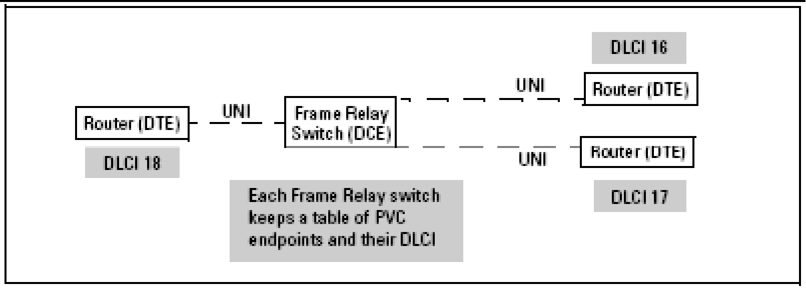


Figure 4 The DLCI Identifies the PVC endpoint.

Multilink Frame Relay

Like Multilink PPP (MLPPP), Multilink Frame Relay (MLFR) aggregates several physical connections into a single logical connection. MLFR helps provide greater access rates for PVCs, particularly in environments in which the greater bandwidth of an E3- or T3-carrier line is not available. MLFR also creates more stable PVCs: If one physical interface goes down, the other interfaces can continue to provide bandwidth for a connection.

Routers that support MLFR FRF.15 bundle multiple carrier lines at the user’s end. The service provider does not recognize the bundle. It assigns each line one Data Link Connection Identifier (DLCI) and carries traffic over the PVC for each line, just as it would for non-bundled lines. The remote router, which also runs MLFR FRF.15, receives the traffic from multiple PVCs but treats it as traffic from a single PVC.

FRF.15 does not require the Frame Relay service provider to support MLFR. However, each bundle can support only one point-to-point connection to a remote site. The remote site must use the same number of carrier lines as the local site.

AOS routers support MLFR FRF.16 rather than FRF.15, which means the service provider must support MLFR. FRF.16 provides several advantages over FRF.15. It allows a bundle to carry multiple PVCs to multiple remote sites, and these sites can use different amounts of bandwidth.

FRF.16 aggregates multiple carrier lines to produce a high-speed connection to the Frame Relay service provider. The service provider supports MLFR, so it recognizes that these lines should be treated as a single bundle. The service provider assigns a DLCI for the lines as a bundle instead of for each line individually. Rather than associating DLCI 101 with E1-carrier line 1 and DLCI 102 with E1-carrier line 2, the Frame Relay service provider associates DLCI 101 and 102 with both E1-carrier lines (See Figure 5).

DLCIs can be requested for PVCs to as many remote sites as an organization needs. In addition, the remote sites do not have to aggregate the same number of lines as the local site or even run MLFR at all. The multilink connection is to the Frame Relay provider, not to the remote sites.

MLFR does not necessarily fragment frames. However, it can use FRF.12 to fragment large frames and minimize delay. An MLFR header is added to the original Frame Relay header to mark the fragments' sequence numbers. In essence, FRF.16 simply increases the committed information rate (CIR) that is negotiated for a Frame Relay port in a T1 or E1 environment.

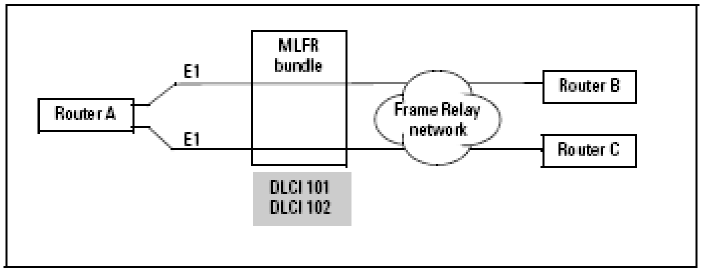


Figure 5 MLFR FRF.16

Figure 5 shows a Frame Relay connection that aggregates two E1-carrier lines to connect to the Frame Relay provider. Router A establishes two PVCs (one to Router B and one to Router C) on this connection.

It is possible to aggregate as many carrier lines as are connected on the AOS Router, as long as they are of equal bandwidth. The MLFR interface can carry as many PVCs as can be requested from the provider. Each PVC draws on the aggregated bandwidth as needed, as available, and in accordance with the CIR negotiated with the service provider. The endpoints of the PVC do not have to use the same number of carrier lines (although a great difference in bandwidth can lead to dropped packets). For example, Router A at the company headquarters can use four E1 lines, while Router C at a small remote site can connect to the network with only one line.

Hardware/Software Requirements/Limitations

Frame Relay was introduced in AOS 1.00.00 and is currently available in the following platforms: NetVanta 1224R (POE), 1224STR (POE), 1335 (POE), 1355/6355, 3200, 3205, 3305, 3430, 3448, 4305, 5305, and 7100. Frame Relay is also available on the Total Access 900 and 900e series Integrated Access Devices (IADs).

Besides being restricted to the above devices, Frame Relay also requires the use of a Network Interface Module (NIM) that supports this technology. The following NIMs support the use of Frame Relay: all T1/E1 NIMs, DDS NIM, Serial NIM, SHDSL (Symmetric High-Bit rate Digital Subscriber Line) NIM, HSSI Wide Interface Module, and DS-3 Wide Interface Module.

Frame Relay Traffic Shaping using CIR and Excess Burst was introduced in 4.01.00 and is currently available in all platforms supported by Frame Relay.

Frame Relay Fragmentation was introduced in 6.01.00 and is currently available in all platforms supported by Frame Relay.

Multilink Frame Relay was introduced in AOS 9.01.00 and is currently available in the following platforms: NetVanta 1224R (POE), 1224STR (POE), 1335 (POE), 1355/6355, 3200, 3205, 3305, 3430, 3448, 4305, 5305, and 7100. Multilink Frame Relay is also available on the Total Access 900e series Integrated Access Devices (IADs).

Configuring Frame Relay in AOS

Frame Relay can be configured within the Web GUI (Graphical User Interface) as well as the CLI (Command Line Interface). The following sections will explain the configuration through the GUI, as well as basic and advanced configuration using the CLI.

Interface configuration mode context	Command	Explanation
t1	• **tdm-group <*number*> timeslots <*range of DS0’s*>** • **coding *[ami/b8zs]*** • **framing *[d4/esf]*** • **clock source *[internal/line/system]*** • **lbo *[long/short]*** • **no shutdown**	• Defines the number of channels (DS0’s) used for the T1 connection • Defines the line coding • Defines the frame format • Defines the clock source or timing for the T1 • Line build out. Sets the strength level of the transmit signal • Activates the interface
e1	• **tdm-group <*number*> timeslots <*range of DS0’s*>** • **coding *[ami/b8zs]*** • **framing *[d4/esf]*** • **clock source *[internal/line/system]*** • **no shutdown**	• Defines the number of channels (DS0’s) used for the E1 connection • Defines the line coding • Defines the frame format • Defines the clock source or timing for the E1 • Activates the interface
dds	• **clock source *[internal/line]*** • **clock rate *[auto/ bps56k/ bps64k]*** • **no shutdown**	• Defines the clock source for the DDS interface • Defines the clock rate • Activates the interface
shdsl	• **equipment-type cpe** • **no shutdown**	• Specifies this unit as a slave unit that is interfacing directly with a service provider • Activates the interface
serial*	• **serial-mode *[EIA530/v35/x21]*** • **et-clock-source *[rxclock/txclock]*** • **no shutdown**	• Configure the interface to support the appropriate cable • Configures the serial interface to take the clock from the receive signal, rxclock, or from the transmit signal, txclock • Activates the interface
t3**	• **clock source *[local/loop]*** • **coding *[b3zs]*** • **framing *[cbit/m13]*** • **line-length *[short/long]*** • **no shutdown**	• Defines the clock source or timing for the T3 • Defines the line coding • Defines the frame format • Sets the strength level of the transmit signal • Activates the interface
hssi**	• **no shutdown**	• Activates the interface

T1, E1, DDS, SHDSL, Serial, T3 and HSSI Interface

The physical T1, E1, DDS, serial, T3, and HSSI interface must be set up and activated in addition to configuring the virtual Frame Relay interface. Table 2 shows the main physical settings that must be configured for an interface that uses Frame Relay.

Table 1. Main physical settings required for T1, E1, DDS, SHDSL, serial, T3 and HSSI interfaces in addition to Frame Relay configuration

* A serial connection on the WAN is typically used when the AOS device is placed behind an existing WAN access device.

** T3 and HSSI are currently only supported in the NetVanta 5305

Configuring Basic Frame Relay Features in CLI

To begin configuring Frame Relay, there must be a logical interface created. Below is the syntax that must be entered from the Global Configuration mode to create a logical main Frame Relay interface:

Syntax: **interface frame-relay** <*interface number*>

Replace <interface number> with any number between 1 and 1024; this number must be unique for each frame relay interface.

Below is an example of what the command prompt may look like once the first frame-relay interface is entered:

```
Router(config)# interface frame-relay 1
Router(config-fr 1)#
```

Once within this mode, entering a "?" will provide configurable options and explanations below the prompt.

```
Router(config-fr 1)#?
```

Interface Configuration Mode Context	**Command**	**Explanation**
frame-relay interface	**No shutdown**	activates the interface
	frame-relay intf-type [dte | dce | nni]	defines the signaling role as user, network, or both
	frame-relay lmi-type [ansi | auto | cisco | none | q933a]	defines Frame Relay signaling type
frame-relay subinterface	**frame-relay interface-dlci** <*dlci*>	defines the DLCI for the PVC
	ip address <A.B.C.D> <subnetmask | /prefix length> or **ip address dhcp** {client-id[Ethernet 0/<port> | HH:HH:HH:HH:HH:HH] | hostname <word>} [no-default-route] [no-domain-name] [no-nameservers] or **ip unnumbered** <interface>	defines a static IP address for the interface configures the Frame Relay subinterface as a DHCP client configures the Frame Relay as an unnumbered interface, which takes its IP address from another interface

global configuration or interface configuration	**cross-connect** <number> <physical interface> <slot>/<port> [<tdm-group number>] **Frame Relay** <interface number>	• cross-connects the physical interface to the logical interface • required tdm-group number for E1 and T1 interfaces (but not for serial interfaces)

Table 2. Frame Relay Configuration Options

The Frame Relay Commands shown above in Table 2 are described in the sections to follow:

Activate the Frame Relay Interface

To bring the Frame Relay interface up, it must be activated administratively. Enter the following command from Frame Relay interface configuration mode context:

Router(config-fr 1)#**no shutdown**

Define the Signaling Role

The signaling role (as explained earlier in the overview) will need to be defined for the AOS router to participate in the Frame Relay network. In most instances, this will be set to the default of DTE (Data Terminal Equipment) which relates to the end user equipment. If the DTE role is used no configuration will be necessary since this is the default behavior.

In some cases, it may be acceptable to modify this setting to test if a certain application will work over a Frame Relay circuit, therefore, signaling may need to be changed to DCE. It is important at the juncture to explain the roles of each type of signaling. If one end of a Frame Relay link is DCE, the other side must be set to DTE to receive and understand the signaling. However, the final type of signaling, NNI, can be set the same on both ends of the circuit. This type of signaling may be used within a point-to-point circuit where there is no intermediate Frame Relay switch (An example of this configuration is provided under Frame Relay Configuration Examples). For leased line circuits where a Frame Relay provider is not utilized PPP should be configured instead of Frame Relay.

To configure the signaling role, enter the following from the Frame Relay interface configuration mode context:

Syntax: **frame-relay intf-type [dce | dte | nni]**

Define the Frame Relay Encapsulation

It is important to mention the Frame Relay encapsulation as it is a setting that can definitely keep a Frame Relay interface from becoming active. AOS routers only support the IETF standard. For inter-vendor compatibility, please verify that the other end of the link is configured to conform to the IETF standard and not a proprietary encapsulation. To configure the Frame Relay encapsulation, use the following command from the Frame Relay interface configuration mode context:

Syntax: **frame-relay encapsulation ietf**

Define the Frame Relay Signaling Type

For a Frame Relay interface to become active, it must match the same signaling type as the other end of the connection from the Frame Relay service provider. From the Frame Relay interface configuration mode context, enter:

Syntax: **frame-relay lmi-type [ansi | auto | cisco | none | q933a]**

Table 3 maps the Frame Relay signaling type to the setting that you must enter for the **frame-relay lmi-type** command.

Signaling Type	**Option**	**Complete Command**
Annex D	**ansi**	**frame-relay lmi-type ansi**
detect signaling type from incoming message	**auto**	**frame-relay lmi-type auto**
Cisco LMI	**cisco**	**frame-relay lmi-type cisco**
no signaling (disables signaling role as well)	**none**	**frame-relay lmi-type none**
Annex A	**q933a**	**frame-relay lmi-type q933a**

Table 3. Frame Relay Signaling

For example, to set the signaling type to **auto**, enter the following command from the Frame Relay interface configuration mode context:

```
Router(config-fr 1)#frame-relay lmi-type auto
```

The default setting for signaling is **ansi**.

Cross-Connecting the Physical Interface to the Logical Interface

On an AOS Router, it is necessary to cross-connect the physical interface to the logical interface so that the router knows which Data Link Layer protocol to use for that WAN connection. After cross-connecting a physical interface to a logical interface, the two are considered a single interface group.

Frame Relay subinterfaces are used to distinguish each PVC using the data-link connection identifier (DLCI). For this reason, it is important to note that physical interfaces are cross-connected to the main Frame Relay interface, and not to individual Fame Relay subinterfaces.

The cross-connect command can be entered from the global configuration mode context or from the main Frame Relay interface configuration mode context, using the following syntax:

Syntax: **cross-connect** <*cross-connect number*> <*physical interface*> <*slot*>/<*port*> [<*tdm-group number*>] **frame-relay** <*logical interface number*>

The <cross-connect number> is globally significant. That is, each **cross-connect** command entered on the router must have a unique cross-connect number.

The <physical interface> parameter needs to be replaced with the appropriate interface type for the corresponding NIM being used. The <slot> and <port> pinpoint this interface's location on the AOS router and distinguish multiple lines of the same type from each other.

If cross-connecting the Frame Relay interface to an T1 or E1 interface, replace <tdm-group number> with the TDM group number defined on the corresponding interface. This option is not needed for other interface types.

The <logical interface number> refers to the number assigned to the configured Frame Relay interface.

For example, to cross-connect the T1 1/1 interface to Frame Relay 1 interface, enter:
Router(config)#**cross-connect 1 t1 1/1 1 fr 1**

If cross-connecting to a serial interface, enter:
Router(config)#**cross-connect 1 ser 1/1 fr 1**

Create the Frame Relay Subinterface

For each frame-relay PVC there needs to be a Frame-Relay subinterface. To create a subinterface, there must first be a main Frame-Relay interface. To create a Frame-Relay subinterface, enter the following command from the Global Configuration context or the main Frame Relay interface configuration mode context:

Syntax: **interface frame-relay** <*number.subinterface number*>
Replace the first number with the main frame-relay interface number. Then replace the subinterface number with any number between 1 and 1007. For simplicity and ease of troubleshooting, it is recommended to make the subinterface number the same as the DLCI (Data-Link Connection Identifier).

For example, if the Frame Relay provider assigned a DLCI of 16 enter:

Router(config-fr 1.16)# **interface frame-relay 1.16**

After entering this mode, the prompt should now be in the Frame Relay subinterface configuration mode context which is reflected in the router prompt:

```
Router(config-fr 1.16)#
```

Once in this menu, additional parameters can be configured including MTU size and Excess Burst Rate. These options are explained in detail in the "Configuring Additional Frame Relay Features in the CLI" section. For initial setup of the Frame-Relay subinterface, only Addressing and DLCI identification are needed.

Assign a DLCI to the Frame Relay Subinterface

The Frame Relay service provider assigns each PVC endpoint a DLCI on the Frame Relay switch, and the switch maintains a table of each DLCI so that it can pass traffic through an outbound port uniquely associated with a specific peer.

To assign the DLCI to the Frame Relay interface, enter the following command from the Frame Relay subinterface configuration mode context:

Syntax: **frame-relay interface-dlci** <DLCI>

Replace <DLCI> with the DLCI that has been provided for this subinterface by the provider. This number will be between 16 and 1007. An important point to note is that every subinterface for a main Frame Relay interface will have to have a different DLCI number.

For example, if a Service Provider assigns one subinterface a DLCI of 16, enter:

Router(config-fr 1.16)# **frame-relay interface-dlci 16**

Configuring the IP Address for the WAN Connection

Since multiple subinterfaces can be configured to one main frame relay interface, the IP addressing must be done on each subinterface rather than the physical interface or the main frame relay interface. There are basically three different ways to assign an IP address on this interface:

- assign a static IP address
- configure the Frame Relay subinterface as a DHCP client
- configure the Frame Relay subinterface as an unnumbered interface

Configuring a Static IP Address

From the Frame Relay subinterface configuration mode context, enter:

Syntax: **ip address** <ip address> <subnet mask | /prefix length>

Replace <ip address> with the IP address that is to be configured for this subinterface. Replace <subnet mask> with the subnet mask or replace </prefix length> with the appropriate CIDR (Classless Interdomain Routing) notation for the mask. For example, an IP address of 192.168.1.1 with a 255.255.255.0 mask would look like the following:

```
Router(config-fr 1.16)# ip address 192.168.1.1 /24
```

Configuring the Frame Relay Subinterface as a DHCP Client

In some circumstances, the provider may assign an IP address for a subinterface dynamically instead of providing a single, static address for the connection. In this situation, it is necessary to assign the subinterface as a DHCP Client to obtain an IP address. Use the following syntax from the Frame Relay subinterface configuration mode context:

Syntax: **ip address dhcp** {client-id [ethernet 0/port] | HH:HH:HH:HH:HH:HH | hostname <word>} [track <name>] [<administrative distance>]

The dhcp command is quite capable in its ability. Table 4 shows the listed options and how they can be used within the dhcp command.

Option	Use	Default Setting
client-id	configures the client identifier displayed in the DHCP server's table	media type and interface's MAC address
hostname	configures the hostname displayed in the DHCP server's table	router hostname
no-default-route	specifies that the DHCP client should not accept the default route obtained through DHCP	accept default route from the server
no-domain-name	specifies that the DHCP clients shoud not accept the domain name included with the other lease settings that the DHCP server sends	accept the domain name setting from the DHCP server
no-nameservers	specifies that the DHCP client should not accept the DNS settings included with the other lease settings that the DHCP server sends	accept DNS settings from the DHCP server
track	attaches a network monitoring track to the DHCP client	(none)
<administrative distance>	specifies the administrative distance to use when adding the DHCP gateway into the route table	1

Table 4. Settings for the DHCP Client Command

It is important to make note of the options in Table 4 before enabling the DHCP client. After enabling the DHCP client, it immediately begins to search for a DHCP server and negotiate a lease. To modify any changes to the lease received from the DHCP server, the lease must run out or be terminated.

Accepting the Default Settings

In most circumstances, it will be easiest to leave the default DHCP settings for the Frame Relay subinterface. To leave the default, enter:

Router(config-fr 1.16)# **ip address dhcp**

The DHCP client on the Frame Relay subinterface will immediately begin to send DHCP discovery message to find a DHCP server. When a DHCP server responds, the client will negotiate an IP address. The DHCP client will send DHCP discovery messages whether or not the Frame Relay subinterface is activated or a valid connection has been established. It will continue to send DHCP discovery messages until a DHCP server responds.

Once the DHCP client has been activated, it is important to make sure that the router actually receives an IP address so that the router is no longer consuming resources and bandwidth by constantly sending out discovery messages on the Frame Relay link. To determine whether the Frame Relay subinterface has an IP address assigned to it, enter the following show command from the enable mode context:

Router(config-fr 1.16)# **do show interface frame-relay** <*number.subinterface number*>

Configuring a Client Identifier

By default, AOS populates the client identifier with the media type and the interface's media access control (MAC) address. The DHCP client can use either the MAC address of a specific Ethernet port or a customized MAC address.

To configure a client identifier when enabling the DHCP client, enter:

Syntax: **ip address dhcp client-id** [Ethernet 0/<*port number*> |
HH:HH:HH:HH:HH:HH]

As the next section shows, it is also possible to configure a hostname with the client identifier.

Configuring a Hostname

AOS uses the hostname configured for the router as the Frame Relay subinterface's default DHCP client hostname. To override this hostname when enabling the DHCP client, enter the following command:

Syntax: **ip address dhcp hostname** <*word*>

For example, to specify that the hostname is *RouterB*, enter the following:

```
Router(config-fr 1.16)# ip address dhcp hostname RouterB
```

The below example shows how to configure not only the hostname, but the client identifier as well:

```
Router(config-fr 1.16)# ip address dhcp client-id ethernet 0/1 hostname RouterB
```

The above command sets the DHCP client to use the MAC address of the Ethernet 0/1 interface as its client identifier and its hostname will be RouterB.

The next three commands described will show how to ignore different settings that are received from the DHCP client.

Overriding Settings received from the DHCP Server

If the DHCP server is configured to provide a default-route, a domain name, or a domain name system (DNS) server, the DHCP client for the Frame Relay subinterface will accept and use these settings. To override this, use one or more of the appropriate options:

Syntax: **ip address dhcp [hostname** <*word*> | no-default-route | no-domain-name | no-nameservers]

For example, to not receive the default route and name server settings through a DHCP client, use the following command:

```
Router(config-fr 1.16)# ip address dhcp no-default-route no-nameservers
```

Attaching a Network Monitoring Tack to the DHCP Interface

As a part of the network monitoring feature, it is possible to attach a network monitoring track to the DHCP client in order to monitor the default route received from a DHCP server. A track uses probes to test routes and servers, with the goal of either removing failed routes or logging poor performance.

Use the **track** option with the **ip address dhcp** command to configure the AOS router to add the default route as a monitored route. The **track** option can be combined with any of the other options for the **ip address dhcp** command (except **no-default-route –** the router cannot monitor a route that the interface does not accept).

For example, to attach the track named DHCPDefault, enter:

```
Router(config-fr 1.16)# ip address dhcp track DHCPDefault
```

Before entering the command, it is important to create a track named DHCPDefault that this command can reference. Also, a probe needs to be created to test the router. For example, the probe could test connectivity to the default gateway listed in the DHCP default route. If the probe fails to reach the gateway, the track determines that the default route has failed and removes it.

> *Tracks require the use of at least AOS Version 13.01.00 to function. The following NetVanta routers have the ability to use this feature: 1335, 3120, 3130, 3305, 3430, 3448, 4305, and the 5305.*

Setting the Administrative Distance

The administrative distance on a DHCP command can be entered to determine if the route received from the DHCP server should be added to the route table. The router uses the administrative distance to determine the best route when multiple routes to the same destination exist. The router assumes that the smaller administrative distance, the more reliable the route is. For the administrative distance to be configured, it cannot be coupled with the **no-default-route** option. Below is the syntax for this command:

Syntax: **ip address dhcp** <*distance*>

Replace <distance> with the administrative distance to add to the default route received. For an administrative distance of 5, enter:

```
Router(config-fr 1.16)#ip address dhcp 5
```

Changing a Setting for the DHCP Client

To change a setting for the DHCP client, the client must first be disabled on that interface. After doing this, enter the command to enable the client with the setting that needs to be changed. Before disabling the client, release the IP address obtained through DHCP. This will prevent the DHCP server from holding the IP address and allow it to assign the IP address to another client.

Releasing or Renewing an IP address

To manually force the Frame Relay subinterface to release or renew an IP address, enter these commands from the Frame Relay subinterface configuration mode context:

```
Router(config-fr 1.16)#ip dhcp release
```

```
Router(config-fr 1.16)#ip dhcp renew
```

Remove the DHCP Client Setting

To disable the DHCP client on a Frame Relay subinterface, use the following command:

```
Router(config-fr 1.16)#no ip address dhcp
```

Configuring the Frame Relay Subinterface as an Unnumbered Interface

To conserve IP addresses it may be necessary to configure the Frame Relay subinterfaces as unnumbered interfaces. If a logical interface has an IP address assigned to it, that address cannot overlap with a subnet that is configured to any other logical interface. As a result, each interface that has an IP address represents a subnet. Depending on the subnetting scheme implemented, this could use more IP address than can be spared.

When assigning a Frame Relay subinterface as an unnumbered interface that uses the IP address assigned to another interface, AOS will use the IP address of the specified interface when sending route updates over the unnumbered interface.

Before configuring the Frame Relay subinterface as an unnumbered interface, be aware of one potential disadvantage: If the interface to which the IP address is actually assigned goes down, the Frame Relay subinterface will be unavailable. For example, suppose Frame Relay 1.16 is configured with an unnumbered interface that takes its IP address from the Ethernet 0/1 interface. If the Ethernet 0/1 interface goes down, the Frame Relay 1.16 subinterface will be unavailable as well.

To minimize the chances of the interface with the IP address going down, assign the IP address to a loopback interface, which typically does not go down.

To configure a Frame Relay subinterface as an unnumbered interface, enter the following command from the Frame Relay subinterface configuration mode context:

Syntax: **ip unnumbered** <*interface*>

Valid interfaces include:

- Ethernet interfaces and subinterfaces
- VLAN interfaces
- Other Frame Relay subinterfaces
- PPP interfaces
- HDLC interfaces
- Loopback interfaces
- Asynchronous Transfer Mode (ATM) subinterfaces
- Demand Interfaces
- BVI interfaces

For example, entering the following commands would configure a loopback interface and then configure the Frame Relay 1.16 subinterface to use the IP address assigned to that loopback interface:

```
Router(config)# interface loopback 1
Router(config-loop 1)# ip address 10.1.1.1 /30
Router(config-loop 1)# interface fr 1.16
Router(config-fr 1.16)# ip unnumbered loopback 1
```

With loopback interfaces, it is unnecessary to use the command **no shutdown** to activate the interface. The status of a loopback interface changes to up once the **interface loopback <*interface number*>** command is entered.

Configuring Frame Relay Traffic Shaping

Setting the CIR

The CIR for a subinterface can be set with the **frame-relay bc** <command> from the Frame Relay subinterface configuration mode context. As explained earlier in the overview, the CIR is the bandwidth that the Frame Relay service provider guarantees for the PVC.

The CIR is calculated from the B_c, which is the maximum number of bits that the Frame Relay carrier guarantees to forward during a certain interval of time (T). The CIR is equal to B_c/T.

It is important to set the B_c for each Frame Relay subinterface to ensure that the PVC does not exceed its CIR. Some Frame Relay service providers may charge a customer if the company consistently transmits over its CIR.

The industry standard is to calculate the time interval as 1 second. As a result, the B_c is essentially the CIR. To set the CIR, enter the following command from the Frame Relay subinterface configuration mode context:

Syntax: **frame-relay bc** <*committed burst value*>

Replace <*committed burst value*> with the CIR expressed in bits. This value can be between 0 and 4,294,967,294 bps.

For example, to set the CIR to be 256 kbps, enter:

```
Router(config-fr 1.16)#frame-relay bc 256000
```

Setting the EIR

In most circumstances where a CIR is negotiated in a SLA, the terms of the agreement probably also allowed for a burst rate on the Frame Relay connection. This burst rate is called the Excess Information Rate (EIR), which defines the maximum amount of traffic your company is allowed to send *in excess* of the CIR.

The B_e sets the maximum number of bits that the router can transmit during “T”. Just as B_c is equal to the CIR, B_e is equal to the EIR. B_e determines the rate at which the AOS Router can burst data above the CIR when there is no congestion on the Frame Relay network.

> *If a value is entered for the **frame-relay bc** command, there should also be a burst rate configured for the Frame Relay link. Otherwise, the link will be limited to the bandwidth specified in the **frame-relay bc** command. Together, the **frame-relay bc** and the **frame-relay be** commands define the amount of bandwidth that can be used on the Frame Relay link. The sum of the values you specify for these two commands should be greater than 8000 but not greater than the port speed at either side of the PVC.*

To configure the EIR for the Frame-Relay subinterface, enter:

Syntax: **frame-relay be** <*excessive burst value*>

Replace <*excessive burst value*> with a burst rate, expressed in bits. You can set a B_e between 0 and 4,294,967,294 bps.

In an instance where the EIR is 64 kbps, enter:

```
Router(config-fr 1.16)frame-relay be 64000
```

Discard Eligible (DE) Bit

After a PVC reaches its CIR, the Frame Relay switch marks each packet with a Discard Eligible (DE) bit. For example, if a PVC's B_c is 1.0 Mb, its B_e is 0.5 Mb, and its transmitting traffic at full capacity, then the Frame Relay switch will set the DE bit on the last 500 kilobytes of packets. If the Frame Relay network becomes congested, the Frame Relay switch first drops the packets that are marked with the DE bit.

Configuring Additional Frame Relay Features in the CLI

In some situations, it is important to configure additional parameters to meet special circumstances. Some of these instances include configuring Frame Relay Multilink, Frame Relay LMI Counters, the MTU value, and Frame Relay Fragmentation.

Configuring Multilink Frame Relay

As mentioned in the Overview, MLFR can be used to aggregate multiple carrier lines into one logical bundle. For configuration, first identify the Frame Relay interface for the connection where an increase in bandwidth is desired. Then, move the configuration mode context for this interface and enable multilink. For example, enter:

```
Router(config)#interface frame-relay 1
Router(config-fr 1)#frame-relay multilink
```

After enabling MLFR on the main Frame Relay interface, it is then possible to cross-connect multiple carrier lines to a Frame Relay interface. On the AOS Router, links are always defined by the Data Link Layer rather than the Physical Layer. By cross-connecting a physical interface to a logical interface, the Data Link Layer protocol is granted access to transmit data over the physical media. This way of defining links makes configuring MLFR easy: simply cross-connect more than one carrier line to the same Frame Relay interface.

At this point, the physical interfaces should have already been configured. If not, see Table 1 which shows the configuration commands needed for a physical interface. To cross-connect the physical interfaces to the Frame Relay interface, the following information is needed:

- Type of carrier line (E1 or T1)
- Network Interface Module slot
- Port number for the interface where the line connects
- TDM group number used for E1 or T1 interfaces

The TDM group number defines the range of channels used by an E1 or T1 carrier line. Lines that will be aggregated can use the same or different TDM group number on each corresponding interface, but these lines *must* use the same number of channels. The TDM group number is only needed for E1 or T1 interfaces. Other interfaces such as serial, dds, SHDSL, etc do not make use of a TDM group.

If a physical interface is already cross-connected to the Frame Relay interface prior to enabling multilink, the existing cross-connect will be removed from the interface. The original line will have to be cross-connected again to the Frame Relay interface along with all the new lines.

The cross-connect command can be entered either from the global or the Frame Relay interface configuration mode context. Enter the command for each carrier line that needs to be bundled:

Syntax: **cross-connect** <cross-connect number> <interface> <slot>/<port> <tdm group number> **frame-relay** <interface number>

Each physical interface should be cross-connected to the Frame Relay interface with a unique cross-connect number. For example, enter:

```
Router(config)#cross-connect 1 t1 1/1 1 fr 1
Router(config)#cross-connect 2 t1 1/2 1 fr 1
Router(config)#cross-connect 3 t1 2/1 1 fr 1
```

> *Physical Interfaces are cross-connected to the Frame Relay* ***interface****, not the Frame Relay* ***subinterface****. This is because Frame Relay subinterfaces define PVCs which are* ***virtual*** *connections, while the Frame Relay interface defines the* ***physical*** *connection available to all the virtual ones.*

MLFR manages the connection by periodically sending out hellos across each carrier line included in the multilink connection. The hello includes the link ID of the line and the bundle ID of the connection as a whole. The link ID lets the router know which lines are up; the bundle ID lets the router know which lines are actually part of the same logical connection.

By default, AOS assigns a bundle ID based on the main Frame Relay interface number in the following format with multilink is utilized:

MFR<*interface number*>

For example, the router might assign the following bundle ID if the Frame Relay interface number is 1:
MFR1

It is possible to configure a bundle ID for the connection manually. Move to the Frame Relay interface configuration mode context and enter:

Syntax: frame-relay multilink bid <*string*>
Replace <string> with the bundle ID.

The bundle ID can be up to 48 characters. It is a good idea to configure the bundle ID at the same time that multilink support is enabled: an active connection will go down briefly and then go back up while the new bundle ID is negotiated. Some carriers will not allow links in a MLFR bundle to come up unless the bundle ID matches what is programmed in their switch. If this is the case the appropriate string must be supplied by the Frame Relay provider and configured on the AOS router.

Configuring Multilink Timers

With MLFR, it is possible to modify different timers related to the hello messages through the Link Integrity Protocol (LIP). AOS Routers allow for configuration of when a bundle link will send out hello messages. The default value is 10 seconds. To configure the hello timer, use the following syntax from Frame Relay interface configuration mode context:

Syntax: **frame-relay multilink hello** <*seconds*>

Replace <seconds> with the number of seconds to wait before sending another hello.

The hello timer can be between 1 and 180 seconds. Remember, that these settings define how often the keepalive will be sent from the AOS router to the provider. It is recommended not to modify this value as it may cause the link to bounce if parameters do not match the other side of the link.

Besides the Hello timer, is is also possible to modify the LIP ACK timer. This defines the amount of time the bundle will wait for an ACK of the first hello message before it resends the hello. The default value is 4 seconds. To configure the ACK timer, use the following from Frame Relay interface configuration mode context:

Syntax: frame-relay multilink ack <*seconds*>
Replace <*seconds*> with the number of seconds to wait for an acknowledgement.

The ACK timer can be between 1 and 10 seconds. Again, this setting modifies an optional setting. Be very careful in modifying this parameter as it will also cause the link to bounce if parameters do not match the other side of the link.

It is also possible to configure the amount of times a Frame Relay bundle will attempt to send a hello message to the far end while waiting for an acknowledgement. The default value is 2 tries. To configure the retry attempts, use the following from Frame-Relay interface configuration mode context:

Syntax: frame-relay multilink retry <*number*>
Replace <*number*> with the number of retries to be sent waiting for an ACK.

The number of retries can be between 1 and 5. This setting, like the others, is completely optional and it is recommended not to modify the default. Since this modifies how often the link sends retries, it is possible that modifying this command could cause instability in the link which could result in the interface bouncing.

Configuring Frame-Relay Multilink Bandwidth Classes

Bandwidth classes are defined in FRF16.1, and place a minimum limit on the acceptable amount of bandwidth required for a bundle to be UP. The classes are defined as follows:

Class A: A single active physical carrier-link is sufficient for the bundle to stay UP.
Class B: All physical carrier-links in the bundle must be active for the bundle to stay UP.
Class C: A minimum threshold of carrier-links must be active for the bundle to stay UP.

The default bandwidth class for all Multilink bundles is A. To modify this parameter, enter the following from the Frame Relay interface configuration mode context:

Syntax: frame-relay multilink bandwidth-class [a | b | [c <*threshold*>]]
Replace threshold with the number of links needed for the bundle to remain active.

The threshold for C can be 1 to 65535, however, any number for the threshold higher than the number of active links cross-connected to the Frame Relay interface will cause the bundle to be in a permanent DOWN state.

Configuring Frame Relay LMI Counters

The Frame Relay counters monitor status polls send and received, track errors, and change the endpoint's signaling status from up to down, depending on the number of errors counted within a set frame of events. Although the counter settings can be tailored to a unique system, most applications do not require the special settings, and therefore should be kept to the default settings.

Table 4 lists the Frame Relay counters, the possible settings, and the polls that each one controls.

Table 4 Frame Relay Counters

Frame Relay Counter	Possible Setting	Default Setting	Description
frame-relay lmi-n391dce <*polls*>	1-255	6	Configure how many link integrity polls in between the full-status polls. Configure this setting for the DCE endpoint.
frame-relay lmi-n391dte <*polls*>	1-255	6	Configure how many link integrity polls occur between the full status polls. Configure this setting for the DTE endpoint.
frame-relay lmi-n392dce <*threshold*>	1-10	3	Configure an error threshold number for the DCE. If the error threshold is met, the signaling status is changed to down, which indicates a service-affecting condition. This condition is cleared after this number of consecutive error-free N392 events are received.
frame-relay lmi-n392dte <*counter*>	1-10	3	Configure an error threshold number for the DTE. If the error threshold is met, the signaling status is changed to down, which indicates a service-affecting condition. This condition is cleared after this number of consecutive error-free N392 events are received.
frame-relay lmi-n393dce <*counter*>	1-10	4	Configure the LMI-monitored event counter for the DCE endpoint.
frame-relay lmi-n393dte <*counter*>	1-10	4	Configure the LMI-monitored event counter for the DTE endpoint.
frame-relay lmi-t391dte <*seconds*>	5-30 seconds	10 seconds	Set the T391 signal-polling timer for the DTE endpoint.
frame-relay lmi-t392dce <*seconds*>	5-30 seconds	10 seconds	Set the T392 polling-verification timer for the DCE endpoint.

No can be added to the beginning of any of these commands to return the counters back to the default settings.

Configuring MTU

The MTU defines the largest size that a frame can be before it must be fragmented. The MTU size on the Frame Relay subinterface should match the MTU used by the remote router and the intervening network devices. Although it is possible to match the MTU on the local Frame Relay interface with that used by the public carrier's equipment, it is almost always not possible to ensure that all the intervening network devices will use the same MTU. To avoid problems that may occur if an intervening network device is using a small MTU size, it is recommended to enable Frame Relay fragmentation (explained below).

> NOTE *If using OSPF routing on the NetVanta router, take special care when setting the MTU. OSPF routers cannot become adjacent if their MTU sizes do not match.*

By default, the MTU for Frame Relay subinterfaces is 1500 bytes. To change this setting, enter the following command from the Frame Relay subinterface configuration mode context:

Syntax: mtu <*size*>
Replace <*size*> with a number between 64 and 1520.

Frame Relay Fragmentation

By default, Frame Relay does not fragment frames as they pass through the network. However, Frame Relay does support a standard under FRF.12 that allows the implementation of fragmentation. It is important to note that for fragmentation to work correctly, frame rate limiting (the CIR) must first be set. It is important that if Frame Relay Fragmentation is to be configured, the provider has also been set up for this setup. To configure fragmentation, enter the following from Frame Relay subinterface configuration mode context:

Syntax: frame-relay fragment <*64..1600*>
Replace <*64..1600*> with the threshold to be used for frame sizes that are candidates for fragmentation. If the value is 1600, it is all values equal to or less than this size, while 64 sets it for all values equal to or greater than this value.

Adding a Description

Descriptions can be added to a Frame Relay interface or subinterface if information needs to be documented for an individual link. For example, multiple PVCs configured on a Frame Relay interface can be documented to show the other end point. In this case, enter the following command at the Frame Relay subinterface configuration mode context:

Syntax: description <*line*>
Replace <*line*> with a phrase up to 80 characters. Below is an example of what might be entered for a PVC:

```
Router(config-fr 1.16)#description WAN link to London office
```

Whenever a **show running-config** command is entered, the description can be displayed. From the enable mode context, enter:

```
Router#show running-config
```

It is also possible to view the description by entering:

```
Router#show running-config interface fr 1.16
```

This command displays the running-config settings for only the Frame Relay 1.16 subinterface, as shown below:

```
interface fr 1.16
  frame-relay interface-dlci 16
  description WAN link to London office
  ip address 192.168.1.1 255.255.255.0
  no shutdown
```

Configuring Frame Relay in the Web GUI

To begin configuration of Frame Relay through the Web Menu, it is important to have the HTTP/HTTPS server activated with a username already configured on the unit. To enable HTTP/HTTPS, enter the following command from the global configuration mode context:

Syntax: ip [http | https] server

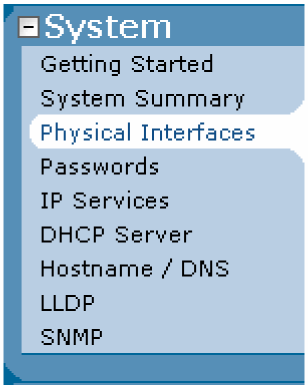
After enabling the GUI Interface, it is then important to add a username and a password for login to the GUI. To do this, enter the following command from the global configuration mode context:

Syntax: username <*username*> password <*password*>
Replace <*username*> and <*password*> with the credentials needed for logging into the GUI.

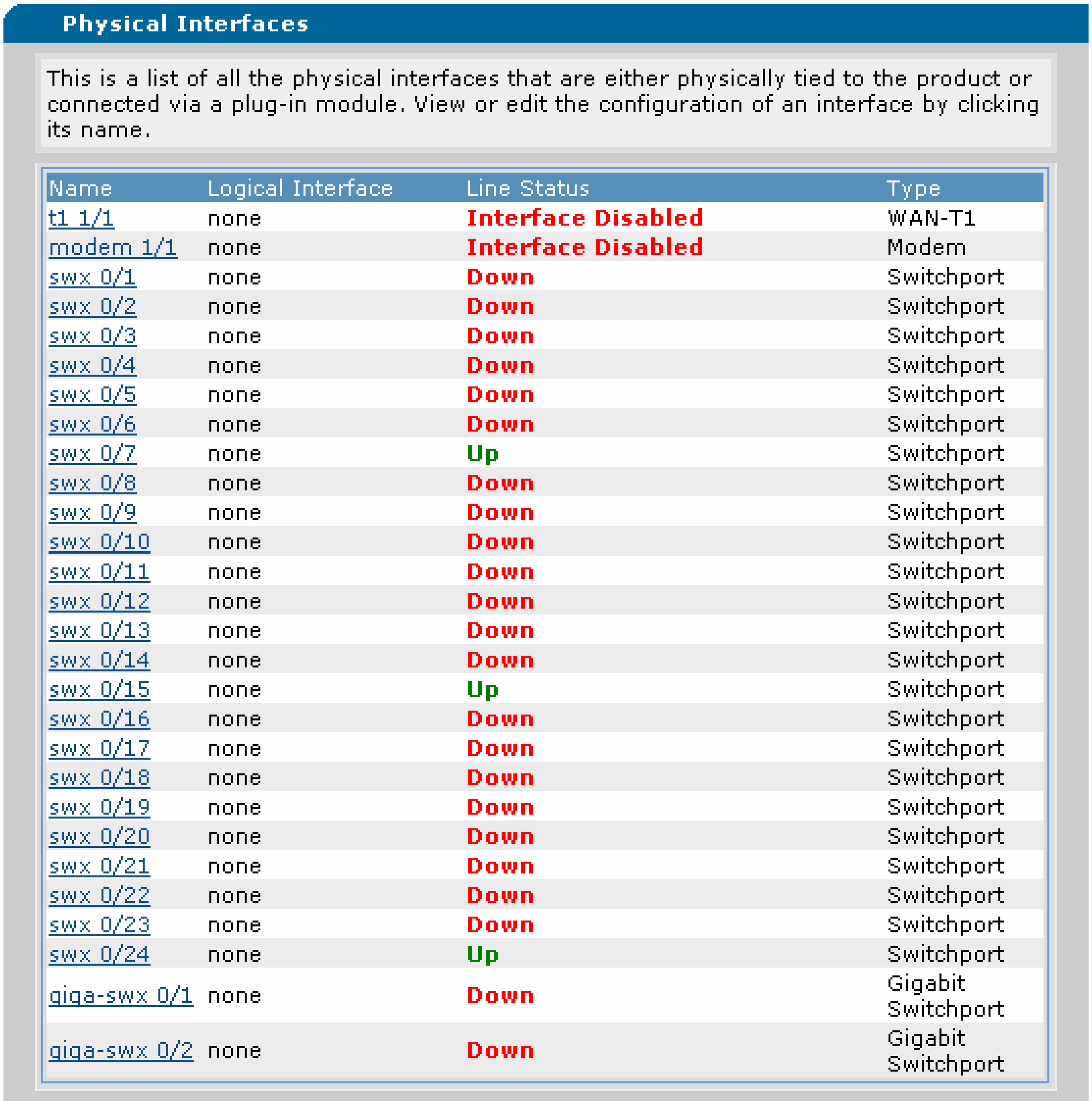
After enabling the server and adding a username, go to any internet browser and enter in the local IP address of the router into the address bar. If you experience difficulties, consult the guide: **Accessing the Web GUI in AOS** available on our knowledgebase at: kb.adtran.com.

Physical Interface Configuration

To begin configuration through the Web GUI, it is first important to have the physical interface set up. From the menu on the left hand side of the screen, choose Physical Interfaces from System Header. Below is an example of the menu option:



On the Physical Interfaces page, there will be choices for each different Physical Interface. Under this menu, choose the interface to be used by Frame Relay. Below, our WAN interface is T1, therefore this is the interface to select.

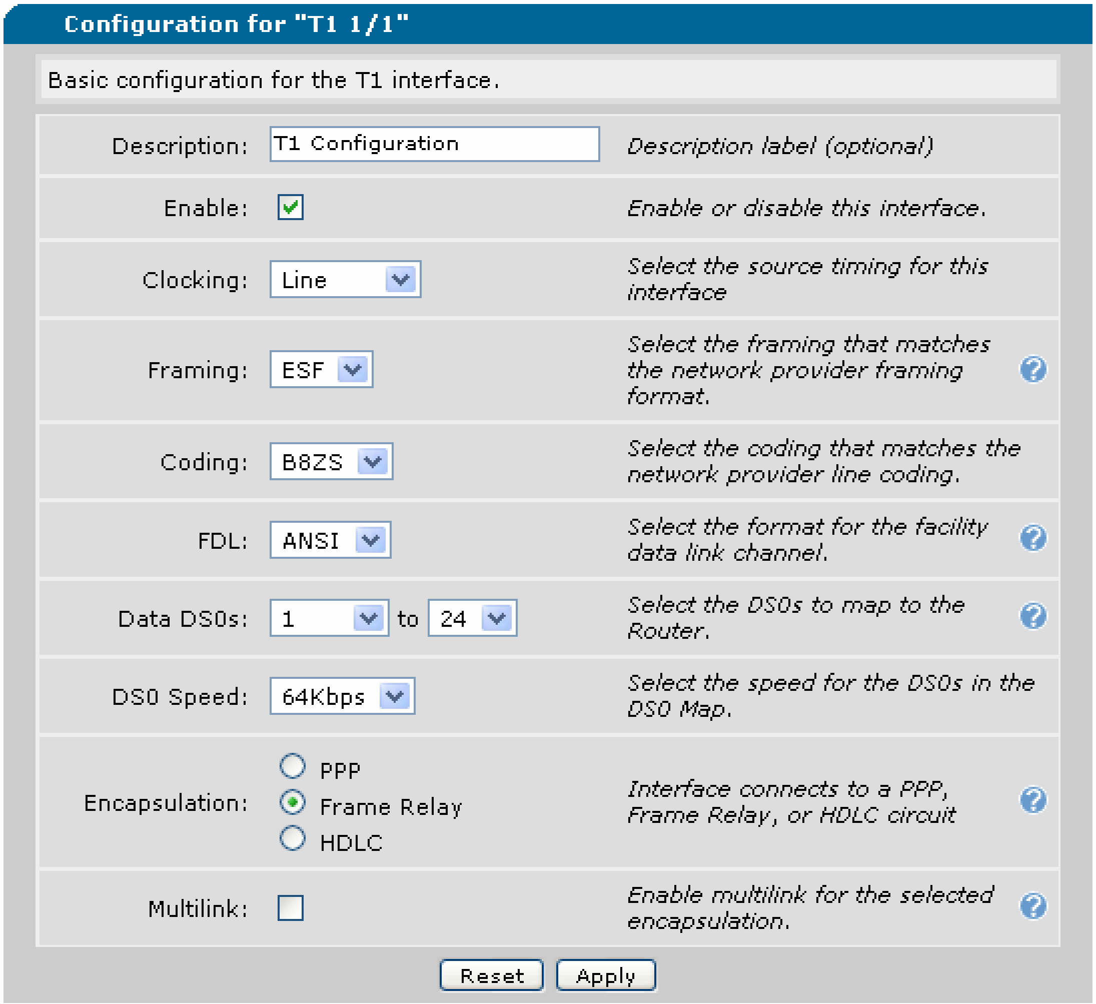


Physical Interfaces

This is a list of all the physical interfaces that are either physically tied to the product or connected via a plug-in module. View or edit the configuration of an interface by clicking its name.

Name	Logical Interface	Line Status	Type
t1 1/1	none	Interface Disabled	WAN-T1
modem 1/1	none	Interface Disabled	Modem
swx 0/1	none	Down	Switchport
swx 0/2	none	Down	Switchport
swx 0/3	none	Down	Switchport
swx 0/4	none	Down	Switchport
swx 0/5	none	Down	Switchport
swx 0/6	none	Down	Switchport
swx 0/7	none	Up	Switchport
swx 0/8	none	Down	Switchport
swx 0/9	none	Down	Switchport
swx 0/10	none	Down	Switchport
swx 0/11	none	Down	Switchport
swx 0/12	none	Down	Switchport
swx 0/13	none	Down	Switchport
swx 0/14	none	Down	Switchport
swx 0/15	none	Up	Switchport
swx 0/16	none	Down	Switchport
swx 0/17	none	Down	Switchport
swx 0/18	none	Down	Switchport
swx 0/19	none	Down	Switchport
swx 0/20	none	Down	Switchport
swx 0/21	none	Down	Switchport
swx 0/22	none	Down	Switchport
swx 0/23	none	Down	Switchport
swx 0/24	none	Up	Switchport
giga-swx 0/1	none	Down	Gigabit Switchport
giga-swx 0/2	none	Down	Gigabit Switchport

After selecting the T1 menu, there will be a set of text boxes where the T1 configuration can be entered. Below is an example of a T1 configuration:



Configuration for "T1 1/1"

Basic configuration for the T1 interface.

Description:	T1 Configuration	*Description label (optional)*
Enable:	☑	*Enable or disable this interface.*
Clocking:	Line	*Select the source timing for this interface*
Framing:	ESF	*Select the framing that matches the network provider framing format.*
Coding:	B8ZS	*Select the coding that matches the network provider line coding.*
FDL:	ANSI	*Select the format for the facility data link channel.*
Data DS0s:	1 to 24	*Select the DS0s to map to the Router.*
DS0 Speed:	64Kbps	*Select the speed for the DS0s in the DS0 Map.*
Encapsulation:	○ PPP ● Frame Relay ○ HDLC	*Interface connects to a PPP, Frame Relay, or HDLC circuit*
Multilink:	☐	*Enable multilink for the selected encapsulation.*

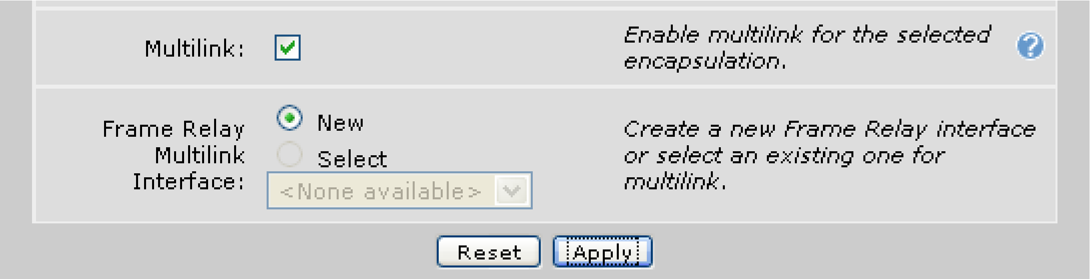
All of these settings match up to CLI commands. Table 5 below shows how each option matches the commands in the CLI:

Table 5: T1 Physical Interface Commands

Option	CLI Command	Explanation
Description	description	This command adds a description to the T1 interface.
Enable	no shutdown/shutdown	Command enables/disables T1 interface
Clocking	clock source [line | internal | through]	Sets the timing for the circuit.
Framing	framing [d4 | esf]	This command sets the framing for the T1 line.
Coding	coding [ami | b8zs]	Sets the bit coding for the T1 line.
FDL	fdl [ansi | att | none]	Configures the FDL

		standard for the T1 line.
Data DSOs and DSO Speed	tdm-group <*number*> timeslots <*range*> speed [56 | 64]	Sets the timeslots of the T1 for Data and the Speed of Each DSO.
Encapsulation	interface [frame-relay | ppp | hdlc] <*interface number*> cross-connect <*cross-connect number*> t1 <*slot*>/<*port*> <*tdm-group number*> [frame-relay | ppp | hdlc] <*interface number*>	Assigns a Layer Two Protocol to the Physical Interface.
Multilink	interface [ppp | frame-relay] <*interface number*> [ppp | frame-relay] multilink	Allows multilink configuration for multiple carrier lines.

If Multilink is set, there will additional options for the Physical Interface. At this point, configuration can be done to set the interface to be a part of an existing multilink bundle or to establish a new bundle as shown below:

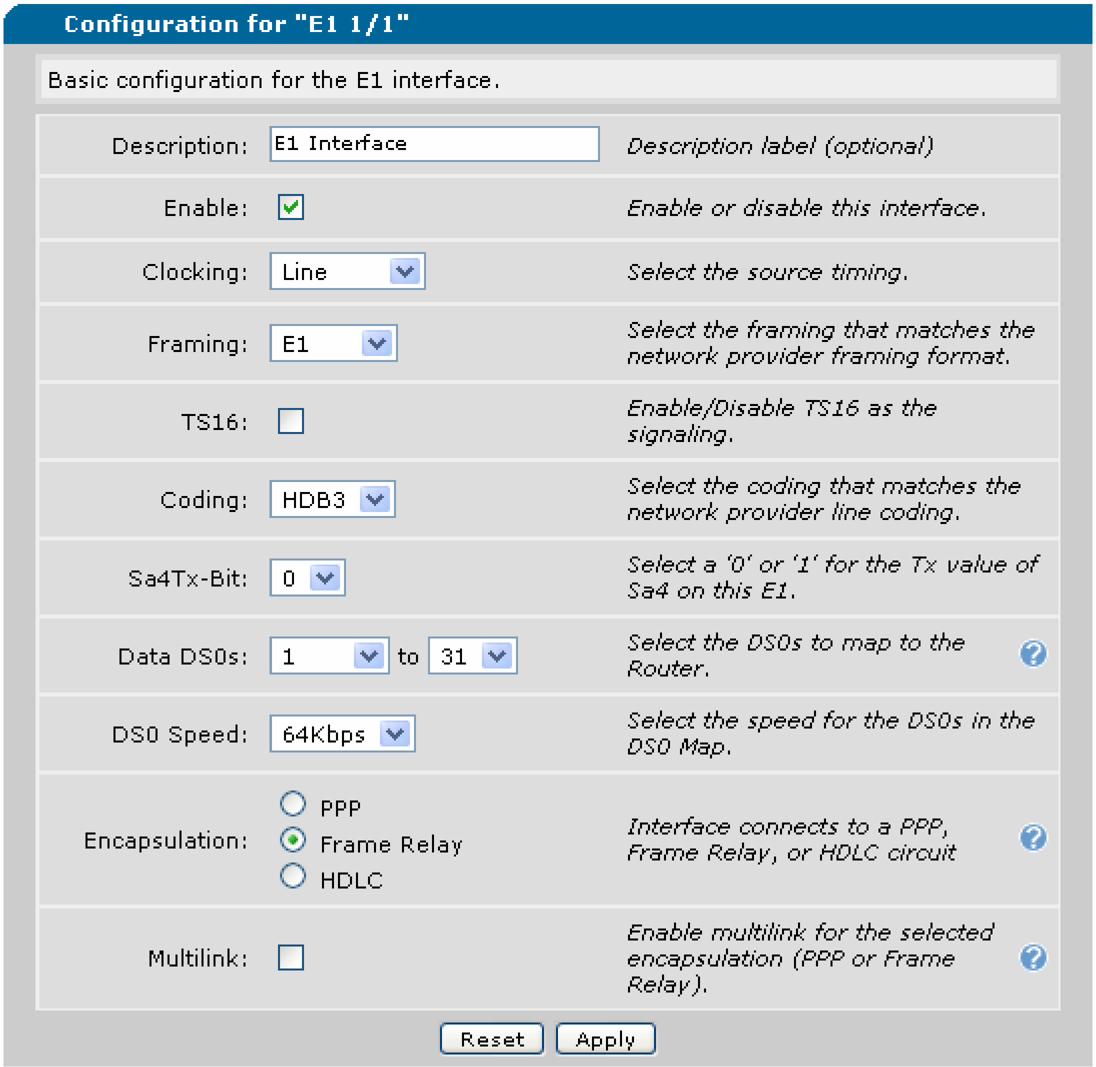


Multilink: ☑ *Enable multilink for the selected encapsulation.*

Frame Relay Multilink Interface: ● New ○ Select <None available> *Create a new Frame Relay interface or select an existing one for multilink.*

At this point, if there was a multilink interface already created, it could be selected instead of creating a new multilink interface. After the physical interface has been configured, click Apply to add the changes to the running-configuration.

Besides a T1, the physical interface may also be E1 or Serial. Below is an example of an E1 configuration. Notice it is very similar to the T1 configuration:



Configuration for "E1 1/1"

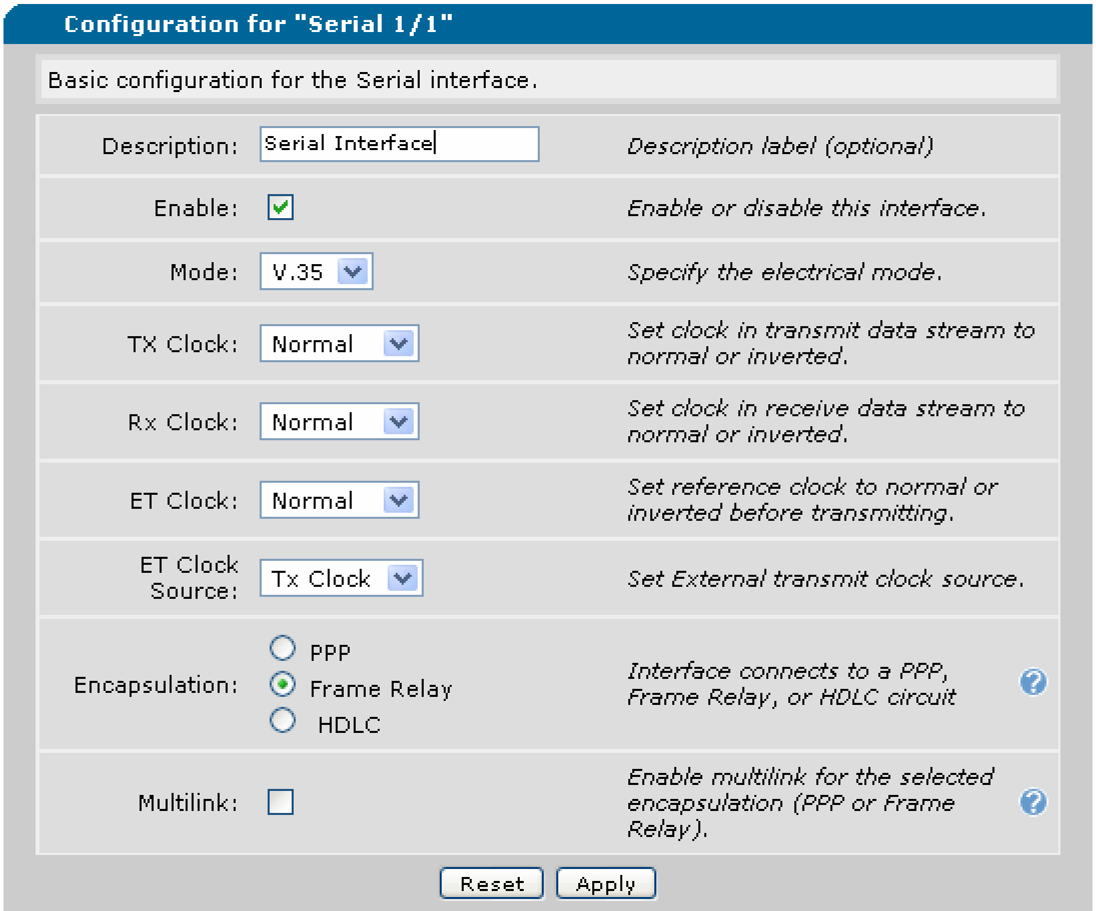
Basic configuration for the E1 interface.

Description:	E1 Interface	*Description label (optional)*
Enable:	☑	*Enable or disable this interface.*
Clocking:	Line	*Select the source timing.*
Framing:	E1	*Select the framing that matches the network provider framing format.*
TS16:	☐	*Enable/Disable TS16 as the signaling.*
Coding:	HDB3	*Select the coding that matches the network provider line coding.*
Sa4Tx-Bit:	0	*Select a '0' or '1' for the Tx value of Sa4 on this E1.*
Data DS0s:	1 to 31	*Select the DS0s to map to the Router.*
DS0 Speed:	64Kbps	*Select the speed for the DS0s in the DS0 Map.*
Encapsulation:	○ PPP ● Frame Relay ○ HDLC	*Interface connects to a PPP, Frame Relay, or HDLC circuit*
Multilink:	☐	*Enable multilink for the selected encapsulation (PPP or Frame Relay).*

As can be seen from the Configuration, there are truly very few differences between E1 and T1. Below is a list of differences:

- Framing can be E1 or CRC4 instead of D4 or ESF
- Signaling can be set to TS16
- Coding can be AMI or HDB3 instead of B8ZS
- The SaTx-Bit can be set to 0 or 1, this is a setting not used at all in T1.

Finally, below is an example of the Physical Interfaces page for a Serial Interface:



Configuration for "Serial 1/1"

Basic configuration for the Serial interface.

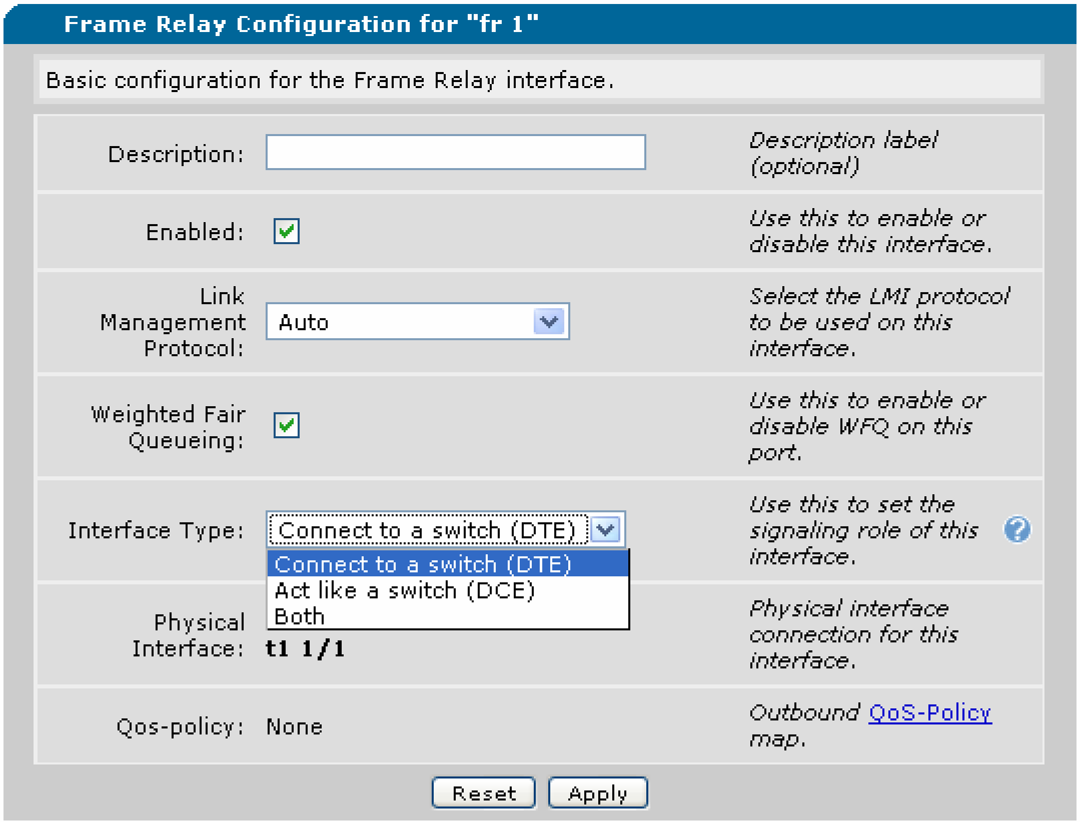
Description:	Serial Interface	*Description label (optional)*
Enable:	☑	*Enable or disable this interface.*
Mode:	V.35	*Specify the electrical mode.*
TX Clock:	Normal	*Set clock in transmit data stream to normal or inverted.*
Rx Clock:	Normal	*Set clock in receive data stream to normal or inverted.*
ET Clock:	Normal	*Set reference clock to normal or inverted before transmitting.*
ET Clock Source:	Tx Clock	*Set External transmit clock source.*
Encapsulation:	○ PPP ● Frame Relay ○ HDLC	*Interface connects to a PPP, Frame Relay, or HDLC circuit*
Multilink:	□	*Enable multilink for the selected encapsulation (PPP or Frame Relay).*

The configuration for Serial 1/1 should be left to the defaults for almost all situations. Below is an example of what the Serial specific commands do:

- Mode: Allows configuration for electrical mode (can be V.35 or X.21)
- TX Clock: Sets the transmit clock to be either normal or inverted.
- RX Clock: Sets the receive clock to be either normal or inverted.
- ET Clock: Sets the reference clock to normal or transmit before transmitting.
- ET Clock Source: Sets external transmit clock source (we could send clocking or receive clocking).

Main Frame Relay Interface Configuration

After configuring, click apply to enter the next menu that allows configuration of the Layer Two Interface. Below is an example of the Frame-Relay page:



Frame Relay Configuration for "fr 1"

Basic configuration for the Frame Relay interface.

Description:		*Description label (optional)*
Enabled:	☑	*Use this to enable or disable this interface.*
Link Management Protocol:	Auto	*Select the LMI protocol to be used on this interface.*
Weighted Fair Queueing:	☑	*Use this to enable or disable WFQ on this port.*
Interface Type:	Connect to a switch (DTE)	*Use this to set the signaling role of this interface.*
Physical Interface:	**t1 1/1**	*Physical interface connection for this interface.*
Qos-policy:	None	*Outbound QoS-Policy map.*

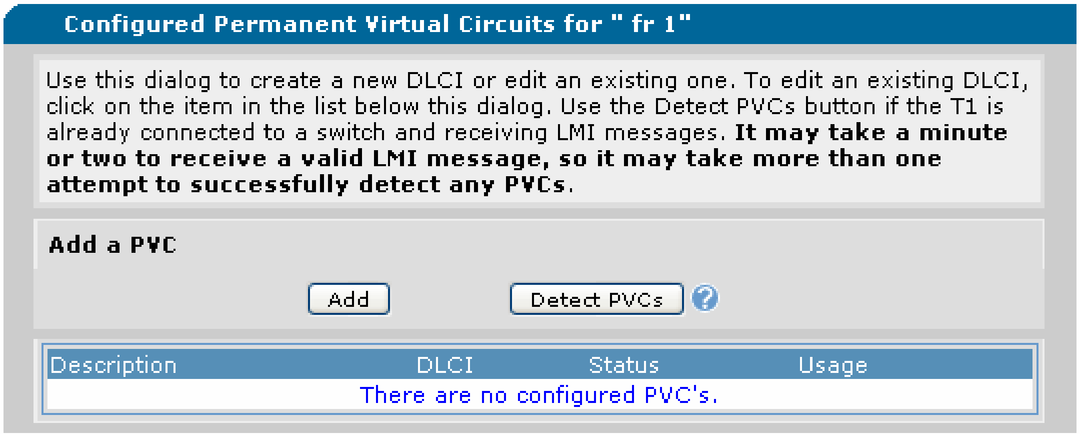
Like the configuration for the physical interfaces, each option above matches a configuration command for the Frame Relay interface configuration mode context. Table 6 shows the configuration commands and how they match up to CLI commands.

Table 6 Frame Relay Main Interface Configuration

Option	CLI Command	Explanation
Description	description	This command adds a description to the Frame Relay interface.
Enabled	no shutdown/shutdown	Enables or Disables the Frame Relay interface.
Link Management Protocol	frame-relay lmi-type [ansi | auto | cisco | none | q933a]	Defines Frame Relay signaling type.
Weighted Fair Queueing	fair-queue	Enables WFQ on interface.
Interface Type	frame-relay intf-type [dte | dce | nni]	Configures Frame Relay interface type.

Frame Relay Subinterface Configuration

After configuring the main frame-relay interface, scroll down the page and there should be a box allowing the addition or auto-detection of PVCs as shown below:



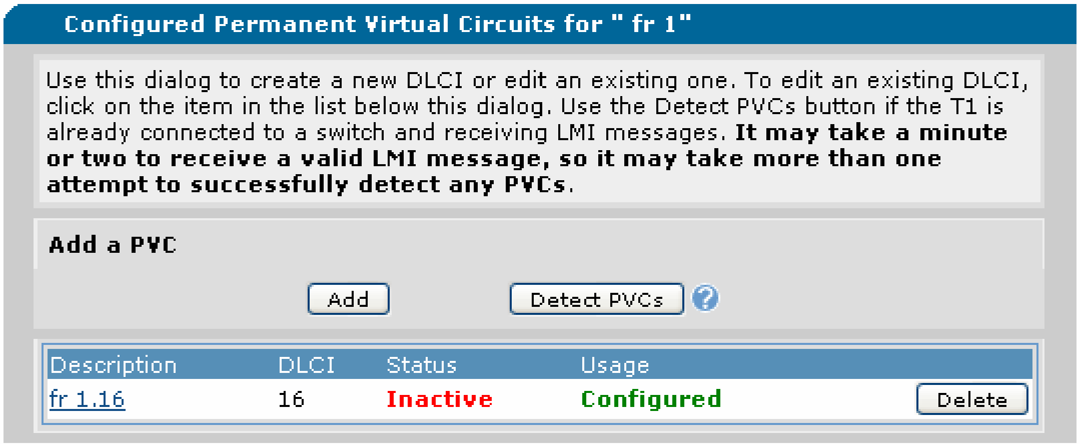
Configured Permanent Virtual Circuits for " fr 1"

Use this dialog to create a new DLCI or edit an existing one. To edit an existing DLCI, click on the item in the list below this dialog. Use the Detect PVCs button if the T1 is already connected to a switch and receiving LMI messages. **It may take a minute or two to receive a valid LMI message, so it may take more than one attempt to successfully detect any PVCs.**

Add a PVC

Description	DLCI	Status	Usage
There are no configured PVC's.			

Click Add to configure a new PVC. Had we already had PVCs configured, it would be shown as below:



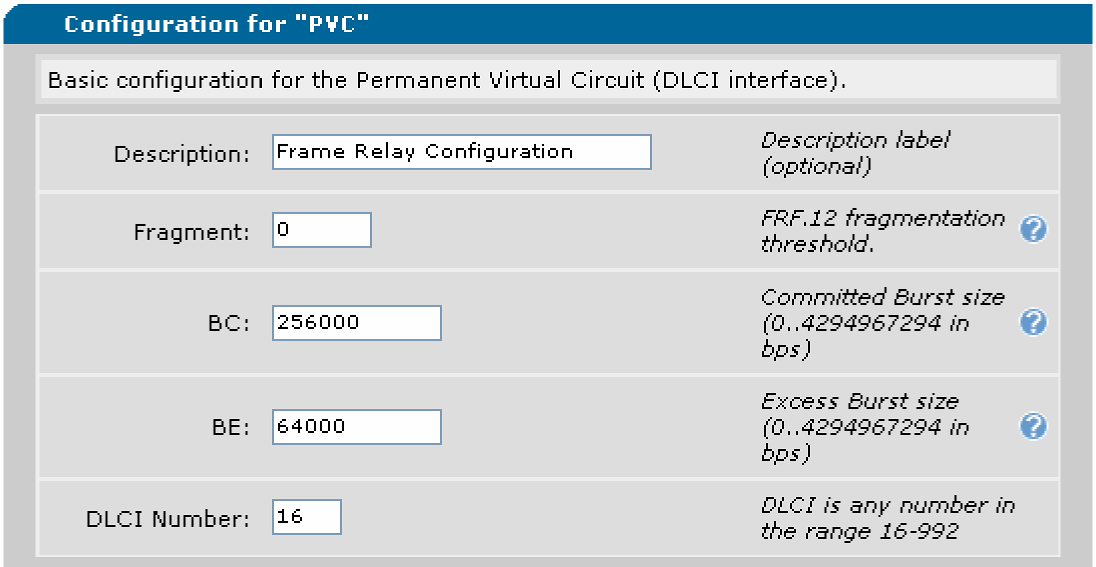
Configured Permanent Virtual Circuits for " fr 1"

Use this dialog to create a new DLCI or edit an existing one. To edit an existing DLCI, click on the item in the list below this dialog. Use the Detect PVCs button if the T1 is already connected to a switch and receiving LMI messages. **It may take a minute or two to receive a valid LMI message, so it may take more than one attempt to successfully detect any PVCs.**

Add a PVC

Description	DLCI	Status	Usage	
fr 1.16	16	**Inactive**	**Configured**	Delete

After clicking add, a new page will appear that allows for configuration of the PVC (Subinterface). Below is an example output from this screen:



Configuration for "PVC"

Basic configuration for the Permanent Virtual Circuit (DLCI interface).

Description:	Frame Relay Configuration	*Description label (optional)*
Fragment:	0	*FRF.12 fragmentation threshold.*
BC:	256000	*Committed Burst size (0..4294967294 in bps)*
BE:	64000	*Excess Burst size (0..4294967294 in bps)*
DLCI Number:	16	*DLCI is any number in the range 16-992*

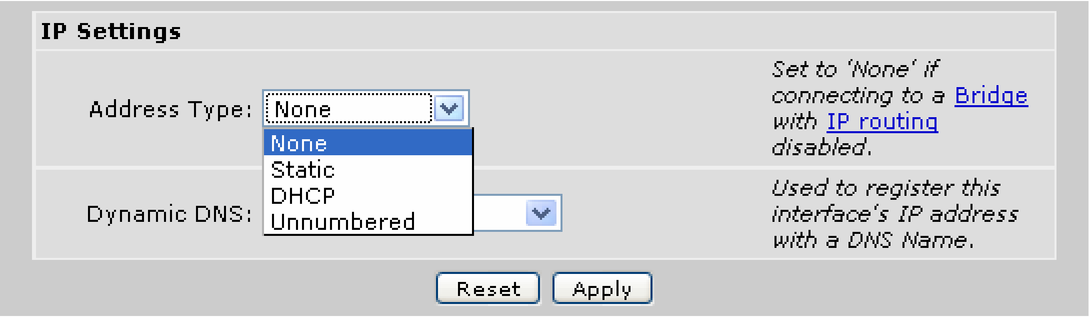
As already stated, each setting matches a configuration command. Table 7 shows how each setting looks within the CLI:

Table 7 Frame Relay Subinterface Configuration

Option	CLI Command	Explanation
Description	description	This command adds a description to the Frame Relay subinterface.
Fragment	frame-relay fragment *<64..1600>*	Configures MTU for frames to be fragmented.
BC	frame-relay bc *<committed burst rate>*	Configures CIR.
BE	frame-relay be *<excess burst rate>*	Configure EIR.
DLCI Number	frame-relay interface-dlci *<dlci>*	Configures DLCI Number.

IP Address Configuration

After configuring the subinterface, scroll down and there will be options for configuring the IP address. There are essentially four different options which are shown below:

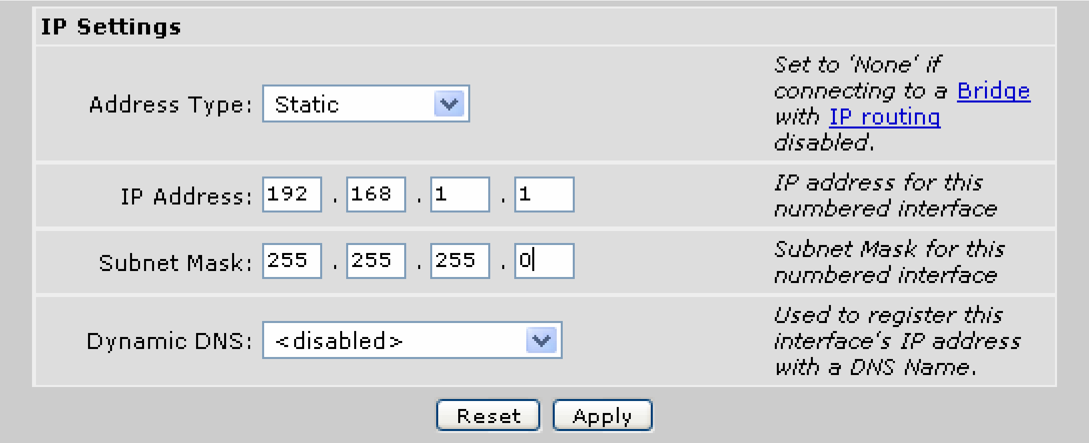


IP Settings

Address Type: None — *Set to 'None' if connecting to a Bridge with IP routing disabled.*

Dynamic DNS: — *Used to register this interface's IP address with a DNS Name.*

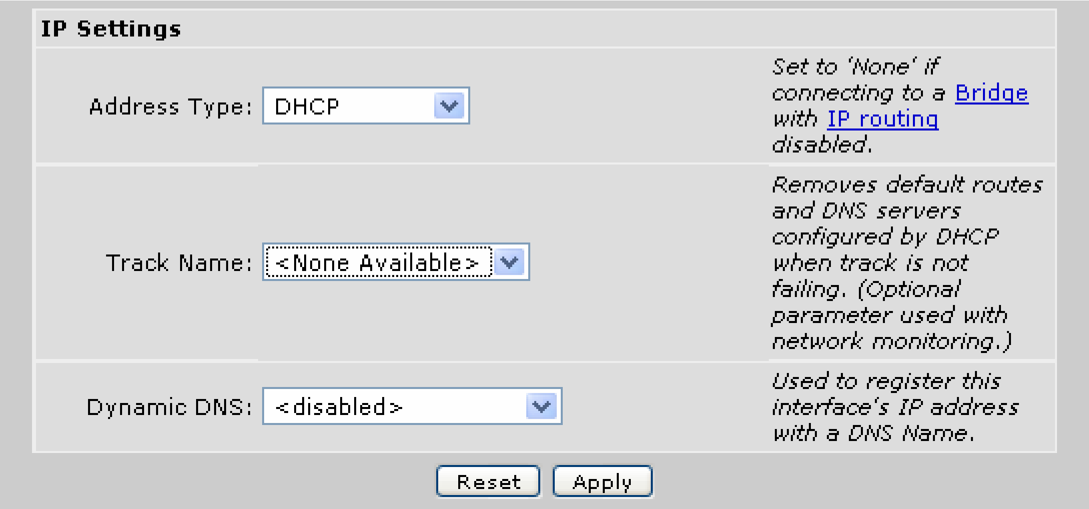
As can be shown, if left to the default, there will not be any IP address on the Frame Relay interface. If one of the other options is selected, more options appear for configuration. Selecting Static will provide the options below:



IP Settings

Address Type:	Static	*Set to 'None' if connecting to a Bridge with IP routing disabled.*
IP Address:	192 . 168 . 1 . 1	*IP address for this numbered interface*
Subnet Mask:	255 . 255 . 255 . 0	*Subnet Mask for this numbered interface*
Dynamic DNS:	<disabled>	*Used to register this interface's IP address with a DNS Name.*

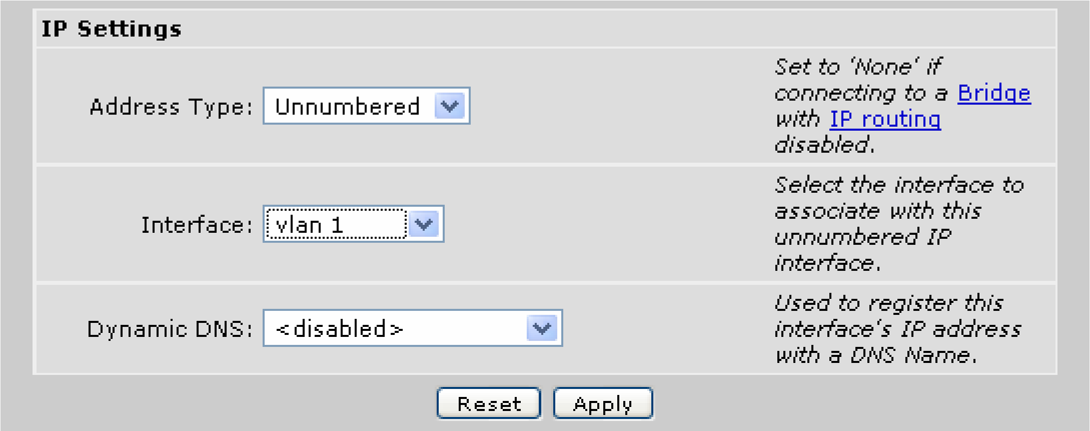
Above shows the information provided by configuring static addressing. This allows the option of configuring both the IP address and the Subnet Mask for the interface. Next, setting DHCP allows the ability to dynamically receive an IP address, this option provides the ability to add a track to the interface if a Default Route is received through DHCP as shown below:



IP Settings

Address Type:	DHCP	*Set to 'None' if connecting to a Bridge with IP routing disabled.*
Track Name:	<None Available>	*Removes default routes and DNS servers configured by DHCP when track is not failing. (Optional parameter used with network monitoring.)*
Dynamic DNS:	<disabled>	*Used to register this interface's IP address with a DNS Name.*

Besides Static, None, and DHCP, it is also possible to configure the interface as Unnumbered. This option also allows the option of which interface the Frame Relay PVC will mimic:



IP Settings

Address Type:	Unnumbered	*Set to 'None' if connecting to a Bridge with IP routing disabled.*
Interface:	vlan 1	*Select the interface to associate with this unnumbered IP interface.*
Dynamic DNS:	<disabled>	*Used to register this interface's IP address with a DNS Name.*

As can be seen, the final option is to configure Dynamic DNS. This option is not needed for Frame Relay configuration, therefore has been omitted from this guide.

Once configuration of the PVC is complete, click apply to set the changes into the running-configuration.

Saving the Configuration

After Applying all Changes for the Frame-Relay interface, it is important to save this configuration. To do this, click on Save in the upper-right hand corner of the screen. The screen below shows what this button will look like:

Example Configurations

Example 1: Point-to-Point Frame-Relay Configuration

Figure 6 shows an Example of a simple point-to-point circuit. It is possible to use Frame-Relay in this application without the use of a Frame-Relay switch.

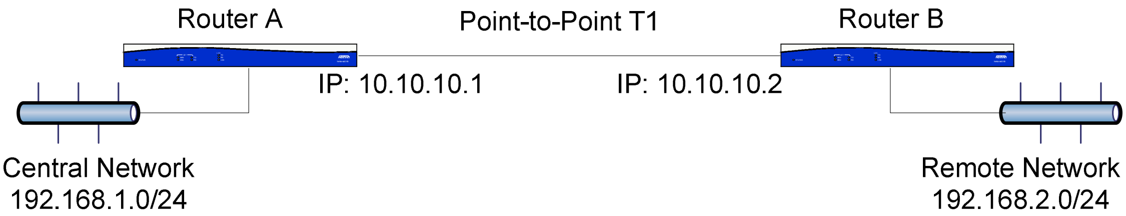


Figure 6 Point-to-Point Network

Below are three different combinations that can be used to configure a Point-to-Point Frame Relay circuit:

Point-to-Point Using DTE and DCE Interface Types

Earlier in the overview and configuration, the interface-type was discussed. The above setup is a scenario that requires the correct interface-type to be configured. Since frame-relay by default expects a Frame-Relay switch on the other end of the link, it is important to make sure that one side is set to DTE (Customer Terminal Equipment) and the other for DCE (Carrier Equipment). Below is an example configuration for both sides of the Frame-Relay circuit:

Router A

```
interface eth 0/1
  ip address 192.168.1.1 255.255.255.0
  no shutdown
!
interface t1 1/1
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
!
interface fr 1 point-to-point
  frame-relay lmi-type ansi
  no shutdown
  cross-connect 1 t1 1/1 1 frame-relay 1
!
interface fr 1.16 point-to-point
  frame-relay interface-dlci 16
  ip address  10.10.10.1  255.255.255.252!
 !
ip route 192.168.2.0 255.255.255.0 10.10.10.2
```

Router B

```
interface eth 0/1
  ip address 192.168.2.1 255.255.255.0
  no shutdown
!
interface t1 1/1
  tdm-group 1 timeslots 1-24
  no shutdown
!
interface fr 1 point-to-point
  frame-relay intf-type dce
  frame-relay lmi-type ansi
  no shutdown
  cross-connect 1 t1 1/1 1 frame-relay 1
!
interface fr 1.16 point-to-point
  frame-relay interface-dlci 16
  ip address  10.10.10.2  255.255.255.252
!
ip route 0.0.0.0 0.0.0.0 10.10.10.1
```

Notice that in Router A, there is no display for the interface-type. This is because the default is DTE.

B. Point-to-Point using NNI Interface Types

Since there is only a slight different between this configuration and the one above, all irrelevant material has been omitted:

Router A

```
interface fr 1 point-to-point
  frame-relay intf-type nni
  frame-relay lmi-type ansi
  no shutdown
  cross-connect 1 t1 1/1 1 frame-relay 1
```

Router B

```
interface fr 1 point-to-point
  frame-relay intf-type nni
  frame-relay lmi-type ansi
  no shutdown
  cross-connect 1 t1 1/1 1 frame-relay 1
```

In the example below, the interface type has been set to NNI. This allows both routers to act as both the Frame-Relay switch and the customer. With this setting, it is important that both sides of the link have been modified to reflect this.

C. Point-to-Point using No LMI

Since LMI takes bandwidth off of the line and in a Point-to-Point scenario there is no need to send this traffic, it is possible to disable LMI on the circuit. Below is an example of this configuration:

Router A

```
interface fr 1 point-to-point
  frame-relay lmi-type none
  no shutdown
  cross-connect 1 t1 1/1 1 frame-relay 1
```

Router B

```
interface fr 1 point-to-point
  frame-relay lmi-type none
  no shutdown
  cross-connect 1 t1 1/1 1 frame-relay 1
```

Example 2 Hub and Spoke Network With Two Remote Sites

Figure 7 is an example Frame Relay network. This setup is quite common when using Frame Relay. In this setup, there is one hub site that actually connects the two remote sites. In this network, the Hub Site has access to each remote site, however, the remotes must send their traffic to the Hub Site to be routed to the next site.

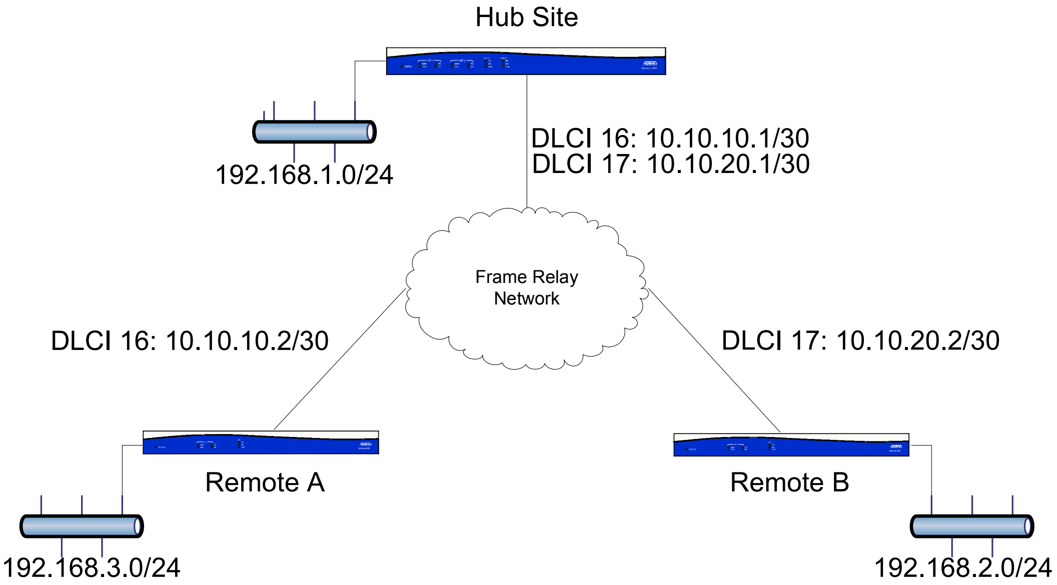


Figure 7 Frame Relay Hub and Spoke Network

For this example, the DLCIs have been set so that both sides of the Frame Relay Network are using the same DLCI as they do at the Hub Site. DLCIs, however, are only locally significant between the Router and the Frame Relay Switch. For this reason, it is possible that the DLCI at the Hub Site that connects to Remote A could be DLCI 40 and the Remote A DLCI number could be DLCI 16.

For the sake of simplicity, it is recommended to match DLCIs with their respective PVCs for each location as has been shown above. In this example, DLCI 16 at the Hub Site has been matched to DLCI 16 at the Remote A site. Notice that the IP Subnet at the Hub Site for DLCI 16 matches the IP Subnet at Remote A for that DLCI. Also, DLCI 17 at the

Hub Site matches DLCI 17 at Remote B as can be shown by the IP Subnets for both of these as well.

Below is the configuration for the Hub Site:

```
interface eth 0/1
  ip address 192.168.1.1 255.255.255.0
  no shutdown
!
interface fr 1 point-to-point
  frame-relay lmi-type ansi
  no shutdown
  cross-connect 1 t1 1/1 1 frame-relay 1
!
interface fr 1.16 point-to-point
  frame-relay interface-dlci 16
  frame-relay bc 512000
  frame-relay be 128000
  ip address  10.10.10.1  255.255.255.252
!
interface fr 1.17 point-to-point
  frame-relay interface-dlci 17
  frame-relay bc 512000
  frame-relay be 128000
  ip address  10.10.20.1  255.255.255.252
!
!
!
!
!
!
!
ip route 192.168.2.0 255.255.255.0 fr 1.16
ip route 192.168.3.0 255.255.255.0 fr 1.17
```

Notice that there are actually two IP interfaces associated with interface fr 1. For this reason, it is possible to route traffic out of either of these two connections. In this example, traffic to 192.168.2.0/24 is being sent out fr 1.16 while 192.168.3.0 is being routed out fr 1.17.

Below is the configuration for Remote A:

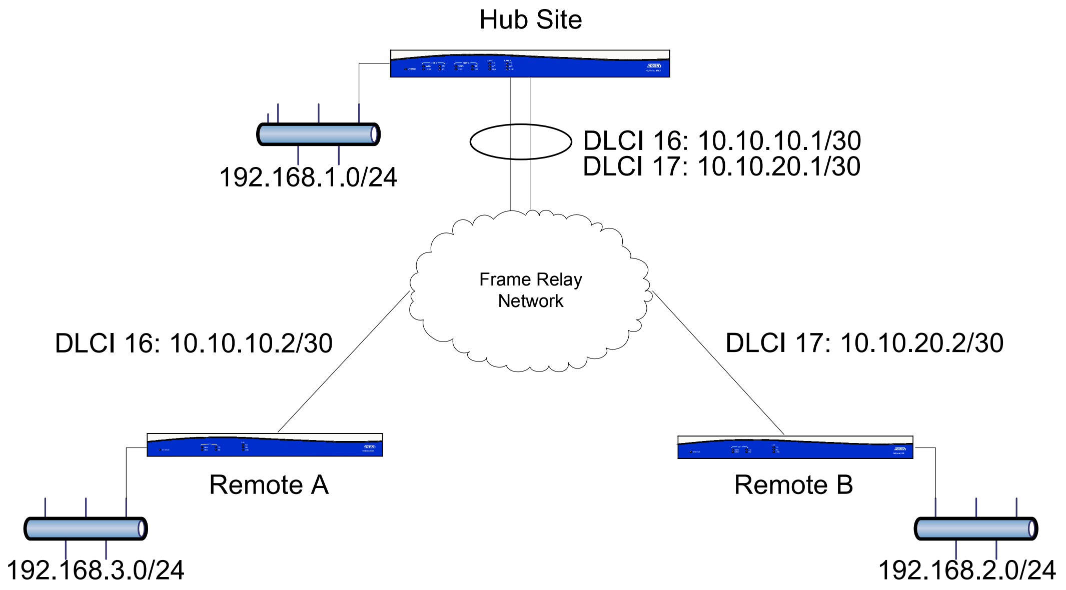
```
interface eth 0/1
  ip address 192.168.2.1 255.255.255.0
!
interface fr 1 point-to-point
```

```
  frame-relay lmi-type ansi
  no shutdown
  cross-connect 1 t1 1/1 1 frame-relay 1
!
interface fr 1.16 point-to-point
  frame-relay interface-dlci 16
  frame-relay bc 512000
  frame-relay be 128000
  ip address  10.10.10.2  255.255.255.252
!
!
!
!
!
!
!
ip route 192.168.1.0 255.255.255.0 fr 1.16
ip route 192.168.3.0 255.255.255.0 fr 1.16
```

Unlike the Hub Site that has two PVCs associated to one interface, the remotes will only have one active PVC which connects each to the main site. Therefore, for Remote A to connect to Remote B, traffic has to be routed through Remote A's active PVC to the Hub Site. This is why there are actually two route statements with an outgoing interface of fr 1.16. For brevity, Remote B's configuration has been omitted.

Example 3 Frame Relay Multilink

Adding on to Example 2, it may be necessary to provide a full T1 of bandwidth between each remote to the main location. In this example, an additional T1 at the Hub Site needs to be added, and the b_c and b_e at each location need to be modified for each PVC. Previously, each PVC was only allowed 512K, however, this site has now upgraded to a full T1 at the remotes and they need a full T1 to the main location. Figure 8 shows the Physical Setup for this configuration:



The oval through the two lines from the Hub Site to the Frame Relay Network shows that there are actually two T1s in a bundle that go to the Frame Relay Network. For brevity, the configurations for Remote A and Remote B have been omitted, however, it is important to note that the b_c and b_e for each location should be modified to match the Hub Site.

Below is the configuration at the Hub Site:

```
interface fr 1 point-to-point
  frame-relay lmi-type ansi
  frame-relay multilink
  no shutdown
  cross-connect 1 t1 3/1 1 frame-relay 1
  cross-connect 2 t1 3/2 1 frame-relay 1
!
interface fr 1.16 point-to-point
  frame-relay interface-dlci 16
  frame-relay bc 1536000
  frame-relay be 8000
  ip address  10.10.10.1  255.255.255.252
!
interface fr 1.17 point-to-point
  frame-relay interface-dlci 17
  frame-relay bc 1536000
  frame-relay be 8000
  ip address  10.10.20.1  255.255.255.252
```

Notice that there has been an additional T1 cross-connected to the frame-relay interface. This has doubled the bandwidth for this logical interface which has in turn allowed the

CIR to be increased at each remote to a single T1 each. Because of this, the b_c for each PVC has been modified to show this increase.

Troubleshooting

Frame Relay provides a substantial amount of debug and show commands that can all be useful when trying to troubleshoot a circuit. This section goes through each of these commands and debugs and then ends with a few troubleshooting scenarios and how these commands can be used to discover the problem. All show and debug commands can be ran from the Privileged Prompt or from any Configuration Mode by implementing the word “do” in front of the command. Below is a list of the show commands, followed by a list of the debug commands for Frame Relay:

Show Commands:

```
show interface frame-relay <port> | <port.sublink>
show frame-relay fragment [interface frame-relay <port.sublink>]
show frame-relay lmi
show frame-relay multilink [dds <slot/port> [detailed] | detailed | e1 <slot/port> [detailed] | fragment [interface frame-relay <port.sublink> | frame-relay <port> [detailed] | serial <slot/port> [detailed] | shdsl <slot/port> [detailed] | t1 <slot/port> [detailed]]
show frame-relay pvc [interface frame-relay <port> [realtime] | realtime]
```

Debug Commands:

```
debug frame-relay events
debug frame-relay llc2
debug frame-relay lmi
debug frame-relay multilink [frame-relay <port> | serial <port/sublink> | states | t1 <slot/port>]
```

Below is example output for each command:

Show Commands

show interface frame-relay 1

```
fr 1 is UP
  Configuration:
    Signaling type is ANSI, signaling role is NETWORK
    Multilink disabled
    Polling interval is 10 seconds,
    full inquiry interval is 6 polling intervals
  Link information:
    5 minute input rate 0 bits/sec, 0 packets/sec
    5 minute output rate 8 bits/sec, 0 packets/sec
    BW 1536 Kbps
```

```
   Queueing method: weighted fair
   HDLC tx ring limit: 2
   Output queue: 0/1/460/64/0 (size/highest/max total/threshold/drops)
     Conversations  0/1/256 (active/max active/max total)
     Available Bandwidth 1152 kilobits/sec
     751 packets input, 43187 bytes
     0 pkts discarded, 0 error pkts, 0 unknown protocol pkts
     928 packets output, 29628 bytes
     0 tx pkts discarded, 0 tx error pkts
```

Most of the information for this command centers around the actual status of the link as well as counters regarding the traffic passing through the link. This command can be helpful in troubleshooting a configuration issue as it lists important information that must match the other end of the Frame-Relay connection. Under the “Configuration” section, it is obvious that this side of the link is configured for Signaling of ANSI (frame-relay lmi-type ansi), the Signaling Role is Network (frame-relay intf-type dce), and multilink is disabled (no fr multilink). The configuration also shows the polling interval and full inquiry intervals.

Next, this command shows the Link Information for the Frame Relay interface. This information includes the input and output rates averaged over a 5 minute time period. This also shows the bandwidth configured on the link, the queuing method, as well as packet input/output counters which includes the discarded and errored packets.

show interface frame-relay 1.16

```
fr 1.16 is Active
  Ip address is 10.10.10.2, mask is 255.255.255.252
  Interface-dlci is 16
  MTU is 1500 bytes, BW is 1536 Kbps
  Average utilization is 0%
  Bc is 1536000 bits
  Be is 8000 bits
```

If the PVC interface is requested, the information provided shows the status of the PVC. In this case, the PVC is active which means that it currently is in use and is operating correctly. As has been stated earlier, if this is *Inactive*, the other side of the link does not have the PVC configured. If the status is *Deleted*, the PVC is deleted on this side of the link while the other side is configured for this DLCI. Other information provided matches the configuration for the PVC including IP address, DLCI, MTU, Bandwidth, and Bc and Be values.

show frame-relay fragment

```
interface dlci frag_size    rx_frag    tx_frag dropped_frag
  fr 1.16   16      1200         0             0             0
```

This command only shows any information if fragmentation has actually been configured on a PVC. The information displayed is a brief look at the fragmentation on the link and shows the size of fragmented frames, how many have been received and transmitted, as well as the number of dropped fragments by the router.

show frame-relay fragment interface frame-relay 1.16

```
DLCI =   16 FRAGMENT SIZE = 1200
rx frag. pkts                      0    tx frag. pkts                  0
rx frag. bytes                     0    tx frag. bytes                 0
rx non-frag. pkts                  3    tx non-frag. pkts              3
rx non-frag. bytes               642    tx non-frag. bytes           603
rx assembled pkts                  3    tx pre-fragment pkts           3
rx assembled bytes               642    tx pre-fragment bytes        603
dropped reassembling pkts          0    dropped fragmenting pkts      0
rx out-of-sequence fragments       0
rx unexpected beginning fragment   0
```

Like the "show frame-relay fragment" command, this shows information only used for an individual PVC if fragmentation has been configured. In this instance, more detailed information is provided including non-fragmented packets and bytes which have been transmitted and received. This information can be very helpful when certain traffic flows seem to have intermittent connectivity or if some applications fail after IP connectivity has been proven.

show frame-relay lmi

```
LMI statistics for interface FR 1 LMI TYPE = ANSI
Num Status Enq. Sent 970      Num Status Msgs Rcvd 463
Num Update Status Rcvd 83     Num Status Timeouts 16
```

This command provides the lmi-type for each frame-relay interface listed. It also provides the number for each type of LMI type that have been sent and received. Among these are how many status inquiries have been sent, how many messages and update statuses have been received. The num status timeouts counts the number of periods a status message was not received.

show frame-relay multilink

```
Bundle: frame-relay 1 is UP; class A bundle
    Near-end BID: MFR1; Far-end BID: MFR1
  Bundle links:
    Link thru t1 5/1 is UP, link state is UP
    Link thru t1 5/2 is UP, link state is UP
```

As shown above, this command shows the status for the multilink bundle, the class of the bundle (A,B, or C, this is documented earlier in the guide), and the Bundle IDs for each side of the link.

show frame-relay multilink detailed

```
Bundle: frame-relay 1 is UP; class A bundle
        Near-end BID: MFR1; Far-end BID: MFR1
  Bundle links:
    Link thru t1 5/1 is UP, link state is UP
      Cause code: none
      Ack timer is 4 seconds, Hello timer is 10 seconds
      Max-retry attempts: 2
      Peer LID is t1 3/1 1, RTT is 2 ms
      Statistics:
        add_link sent: 4, add_link received: 3
        add_link ack sent: 3, add_link ack received: 4
        add_link rej sent: 0, add_link rej received: 0
        remove_link sent: 2, remove_link received: 0
        remove_link ack sent: 0, remove_link ack received: 2
        hello sent: 142, hello received: 142
        hello ack sent: 142, hello ack received: 142
    Link thru t1 5/2 is UP, link state is UP
      Cause code: none
      Ack timer is 4 seconds, Hello timer is 10 seconds
      Max-retry attempts: 2
      Peer LID is t1 3/2 1, RTT is 2 ms
      Statistics:
        add_link sent: 4, add_link received: 3
        add_link ack sent: 3, add_link ack received: 4
        add_link rej sent: 0, add_link rej received: 0
        remove_link sent: 2, remove_link received: 0
        remove_link ack sent: 0, remove_link ack received: 2
        hello sent: 141, hello received: 140
        hello ack sent: 140, hello ack received: 141
```

This command shows detailed information provided by LMI. This information includes add and remove link updated that have been sent and received as well as hellos and hello acknowledgements that have been sent and received through LMI for each link.

show frame-relay multilink dds 1/1 | t1 1/1 | shdsl 1/1 | serial 1/1 | e1 1/1

```
Bundle-link thru dds 1/1 is DOWN, link state is DOWN
On bundle frame-relay 1, BID: MFR1
```

All of the above commands will display similar information to the above. Basically, this command will show the output for one single link within the multilink bundle.

show frame-relay multilink dds 1/1 detailed | t1 1/1 detailed | shdsl 1/1 detailed | e1 1/1 detailed

```
Bundle-link thru t1 5/1 is UP, link state is UP
On bundle frame-relay 1, BID: MFR1
  Cause code: none
  Ack timer is 4 seconds, Hello timer is 10 seconds
  Max-retry attempts: 2
  Peer LID is t1 3/1 1, RTT is 2 ms
  Statistics:
    add_link sent: 4, add_link received: 3
    add_link ack sent: 3, add_link ack received: 4
    add_link rej sent: 0, add_link rej received: 0
    remove_link sent: 2, remove_link received: 0
    remove_link ack sent: 0, remove_link ack received: 2
    hello sent: 121, hello received: 120
    hello ack sent: 120, hello ack received: 121
```

This command takes the detailed command from above and cuts the information down to a single physical link instead of showing all traffic for all interfaces.

show frame-relay multilink fragment interface frame-relay 1.16

```
DLCI =   16 FRAGMENT SIZE =    0
rx frag. pkts                      0  tx frag. pkts                  0
rx frag. bytes                    55  tx frag. bytes                 0
rx non-frag. pkts                 61  tx non-frag. pkts             57
rx non-frag. bytes             11153  tx non-frag. bytes         11064
rx assembled pkts                  0  tx pre-fragment pkts          57
rx assembled bytes                 0  tx pre-fragment bytes      10950
dropped reassembling pkts          0  dropped fragmenting pkts      0
rx assembly timeouts              54
rx out-of-sequence pkts held      55
rx unexpected beginning fragment   6
```

This command acts much like the “show frame-relay fragment interface frame-relay 1.16, however it adds an extra field that shows the number of assembly timeouts that have been received by this router.

show frame-relay multilink frame-relay 1

This command is an exact duplicate of the information from a “show frame-relay multilink,” however, it will only show the bundle for frame-relay 1 if multiple bundles are configured on one device.

show frame-relay multilink frame-relay detailed

This command is an exact duplicate of the information from a “show frame-relay multilink detailed,” however, it will only show the bundle for frame-relay 1 if multiple bundles are configured on one device.

show frame-relay pvc

```
Frame Relay Virtual Circuit Statistics for interface FR 1
          Active    Inactive   Deleted   Static
  local      1         0          0         1

DLCI = 16, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = FR
1.16
  MTU: 1500
  input pkts:   74      output pkts:  76       in bytes:     14219
  out bytes:    14617    dropped pkts: 1        in FECN pkts: 0
  in BECN pkts: 0        in DE pkts:   0        out DE pkts:  0
  Creation time: 01-26-2000 00:35:47   Last status change: 00W:00D:00H:33M:18S
```

This command shows each individual PVC including the DLCI, the status, the interface name, and the MTU. It also shows the number of Active, Inactive, Deleted, and Statically configured PVCs on the box. This command can be extremely effective for other reasons, however, as it shows updates related to congestion. As can be shown, there are counters that show how many BECNs (Backward Explicit Congestion Notifications) and FECNs (Forward Explicit Congestion Notifications). These two counters are very important as they display if any traffic is being dropped due to overcongestion on the line.

If the router receives a BECN, then it knows that it is transmitting too much information. If this occurs consistently, it may be time to take a look at the b_c and b_e. If it happens that FECNs are being received, then the remote side is transmitting too much information. In this case, the Frame Relay switch is notifying the receiving side that congestion is occurring before the traffic reaches its destination. In the same way as with BECNs, the b_c and b_e need to be checked on the far end.

Debug Commands

These debug commands were all taken from when the links are brought up:

debug frame-relay events

```
Router(config-fr 1)#shutdown
2000.01.26 01:35:39 FRAME_RELAY.LMI Port 1 vc 16: state changed to Inactive
2000.01.26 01:35:39 FRAME_RELAY.LMI fr 1 FRE Hardware not active

2000.01.26 01:35:39 FRAME_RELAY.MFR LIP Link t1 5/2 1 : state changed to DOWN
2000.01.26 01:35:39 FRAME_RELAY.MFR LIP Link t1 5/1 1 : state changed to DOWN
2000.01.26 01:35:39 INTERFACE_STATUS.fr 1 changed state to administratively down
2000.01.26 01:35:39 INTERFACE_STATUS.fr 1 changed state to down
2000.01.26 01:35:39 INTERFACE_STATUS.MFR1 changed state to administratively
down
2000.01.26 01:35:39 INTERFACE_STATUS.MFR1 changed state to down
2000.01.26 01:35:39 FRAME_RELAY.LMI fr 1 FRE Hardware active
Router(config-fr 1)#no shutdown
2000.01.26 01:35:44 INTERFACE_STATUS.fr 1 changed state to administratively up
2000.01.26 01:35:44 INTERFACE_STATUS.MFR1 changed state to administratively up
2000.01.26 01:35:45 INTERFACE_STATUS.MFR1 changed state to up
2000.01.26 01:35:49 FRAME_RELAY.MFR LIP Link t1 5/2 1 : state changed to UP
2000.01.26 01:35:49 FRAME_RELAY.MFR LIP Link t1 5/1 1 : state changed to UP
2000.01.26 01:35:51 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive because the
port is down.
2000.01.26 01:35:51 FRAME_RELAY.LMI Port 1 vc 16: The switch reports VC 16 as
inactive.
2000.01.26 01:35:51 FRAME_RELAY.LMI fr 1 coming up with last good Rx seq# 44
2000.01.26 01:35:52 INTERFACE_STATUS.fr 1 changed state to up
2000.01.26 01:36:01 FRAME_RELAY.EVENTS fr 1.16 (tx): IARP request sent
2000.01.26 01:36:11 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive due to DTE
signaling.
2000.01.26 01:36:11 FRAME_RELAY.LMI Port 1 vc 16: The switch reports VC 16 as
active.

2000.01.26 01:36:11 FRAME_RELAY.LMI Port 1 vc 16: state changed to Active
2000.01.26 01:36:15 FRAME_RELAY.EVENTS fr 1.16 (tx): IARP request sent
2000.01.26 01:36:15 FRAME_RELAY.EVENTS fr 1.16 (rx): IARP response from
10.10.10.2
```

This debug updates on general information regarding the link. In this case, the information shown revolves around a frame-relay interface being dropped and then brought back up. Some of the output included is the time and date for each instance, what process was involved in the event (i.e. FRAME_RELAY.MFR LIP), and the event that occurred. This information is a little more helpful than event messages that appear on the screen as it constantly is updated.

debug frame-relay llc2

debug frame-relay lmi

```
2000.07.30 20:09:43 FRAME_RELAY.LMI fr 1 DTE: Full Status Inquiry

2000.07.30 20:09:43 FRAME_RELAY.LMI fr 1 Tx LMI local seq: 63 remote seq: 2

2000.07.30 20:09:44 FRAME_RELAY.LMI fr 1 Rx LMI local seq: 63 remote seq: 3

2000.07.30 20:09:44 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive because the
port is down.

2000.07.30 20:09:44 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive due to DTE
signaling.

2000.07.30 20:09:44 FRAME_RELAY.LMI Port 1 vc 16: The switch reports VC 16 as
inactive.

2000.07.30 20:09:44 FRAME_RELAY.LMI Port 1 vc 16: DTE signaling taking down
VC 16

2000.07.30 20:09:53 FRAME_RELAY.LMI fr 1 DTE: Full Status Inquiry

2000.07.30 20:09:53 FRAME_RELAY.LMI fr 1 Tx LMI local seq: 64 remote seq: 3

2000.07.30 20:09:54 FRAME_RELAY.LMI fr 1 Rx LMI local seq: 64 remote seq: 4

2000.07.30 20:09:54 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive because the
port is down.

2000.07.30 20:09:54 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive due to DTE
signaling.

2000.07.30 20:09:54 FRAME_RELAY.LMI Port 1 vc 16: The switch reports VC 16 as
inactive.

2000.07.30 20:09:54 FRAME_RELAY.LMI Port 1 vc 16: DTE signaling taking down
VC 16

2000.07.30 20:10:03 FRAME_RELAY.LMI fr 1 DTE: Full Status Inquiry

2000.07.30 20:10:03 FRAME_RELAY.LMI fr 1 Tx LMI local seq: 65 remote seq: 4

2000.07.30 20:10:04 FRAME_RELAY.LMI fr 1 Rx LMI local seq: 65 remote seq: 5

2000.07.30 20:10:04 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive because the
port is down.
```

```
2000.07.30 20:10:04 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive due to DTE
signaling.

2000.07.30 20:10:04 FRAME_RELAY.LMI Port 1 vc 16: The switch reports VC 16 as
inactive.

2000.07.30 20:10:04 FRAME_RELAY.LMI Port 1 vc 16: DTE signaling taking down
VC 16

2000.07.30 20:10:04 FRAME_RELAY.LMI fr 1 coming up with last good Rx seq# 5

2000.07.30 20:10:04 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive due to DTE
signaling.

2000.07.30 20:10:04 FRAME_RELAY.LMI Port 1 vc 16: The switch reports VC 16 as
active.

2000.07.30 20:10:04 FRAME_RELAY.LMI fr 1 FRE Rx async status msg

2000.07.30 20:10:04 FRAME_RELAY.LMI Port 1 vc 16: DTE signaling wants to bring
up VC 16

2000.07.30 20:10:04 FRAME_RELAY.LMI Port 1 vc 16: state changed to Active
```

As can be seen, this information is regarding the messages that are being sent and received through the use of LMI. Besides just showing this, it also provides status changes for each PVC. To know what interface each message is applied to, it is important to look at the Port listed after FRAME_RELAY.LMI. This with the addition of vc 16 lets you know these messages are for DLCI 16 on port 1 (interface fr 1.16).

debug frame-relay multilink

```
2000.02.01 19:47:03 FRAME_RELAY.MULTILINK (O): msg=ADD_LINK, Link=t1
5/1 1, Bundle=MFR1, BL state=IDLE
2000.02.01 19:47:03 FRAME_RELAY.MULTILINK (O): msg=ADD_LINK, Link=t1
5/2 1, Bundle=MFR1, BL state=IDLE

2000.02.01 19:47:03 FRAME_RELAY.LMI Port 1 vc 16: state changed to
Active2000.02.01 19:47:03 FRAME_RELAY.MULTILINK (I):
msg=ADD_LINK_ACK, Link=t1 5/1 1, Bundle=MFR1, BL state=ADD_SENT
2000.02.01 19:47:03 FRAME_RELAY.MULTILINK (I): msg=ADD_LINK_ACK,
Link=t1 5/2 1, Bundle=MFR1, BL state=ADD_SENT
Router(config-fr 1)#
2000.02.01 19:47:03 INTERFACE_STATUS.fr 1 changed state to administratively up
```

```
2000.02.01 19:47:03 INTERFACE_STATUS.MFR1 changed state to administratively
up2000.02.01 19:47:04 FRAME_RELAY.MULTILINK (I): msg=ADD_LINK, Link=t1
5/2 1, Bundle=MFR1, BL state=ACK_RX
2000.02.01 19:47:04 FRAME_RELAY.MULTILINK (O): msg=ADD_LINK_ACK,
Link=t1 5/2 1, Bundle=MFR1, BL state=ACK_RX

2000.02.01 19:47:04 FRAME_RELAY.MFR LIP Link t1 5/2 1 : state changed to
UP2000.02.01 19:47:04 FRAME_RELAY.MULTILINK (I): msg=ADD_LINK, Link=t1
5/1 1, Bundle=MFR1, BL state=ACK_RX
2000.02.01 19:47:04 FRAME_RELAY.MULTILINK (O): msg=ADD_LINK_ACK,
Link=t1 5/1 1, Bundle=MFR1, BL state=ACK_RX

2000.02.01 19:47:04 FRAME_RELAY.MFR LIP Link t1 5/1 1 : state changed to UP
2000.02.01 19:47:04 INTERFACE_STATUS.fr 1 changed state to up
2000.02.01 19:47:04 INTERFACE_STATUS.MFR1 changed state to up
```

This information basically shows when which messages and for which link the messages have been sent to be added to the Frame Relay interface. After the Frame Relay interface comes up, there is are two messages sent requesting the links be added to the Bundle MFR1. Those two links are t1 5/1 and t1 5/2. After this, the far end responds with an acknowledgement. Within these debug messages, there is a title of the debug message (FRAME_RELAY.MULTILINK), after that, there is the letter "I" or "O" listed in parenthesis. This letter represents the direction the debug message is for. "O" would represent a message being sent by the router, while "I" represents a message incoming to the router. Once the links have been added and the acknowledgements have been sent and received for each link, the debug command stabilizes and then only displays debugs for each LMI Hello message as shown below:

```
2000.02.01 19:52:54 FRAME_RELAY.MULTILINK (I): msg=HELLO, Link=t1 5/1 1,
Bundle=MFR1, BL state=UP
2000.02.01 19:52:54 FRAME_RELAY.MULTILINK (O): msg=HELLO_ACK, Link=t1
5/1 1, Bundle=MFR1, BL state=UP
```

debug frame-relay multilink frame-relay <*port*>

This command works the same as the command above (debug frame-relay multilink), however, this command will only show the output for a debug of this one multilink bundle, instead of all frame-relay multilink bundles on the router.

debug frame-relay multilink states

```
2000.02.01 19:56:48 FRAME_RELAY.MULTILINK Link t1 5/1 1 state change: old
state=IDLE, new state=ADD_SENT
2000.02.01 19:56:48 FRAME_RELAY.MULTILINK Link t1 5/2 1 state change: old
state=IDLE, new state=ADD_SENT
```

```
2000.02.01 19:56:48 FRAME_RELAY.MULTILINK Link t1 5/1 1 state change: old
state=ADD_SENT, new state=ACK_RX
2000.02.01 19:56:48 FRAME_RELAY.MULTILINK Link t1 5/2 1 state change: old
state=ADD_SENT, new state=ACK_RX
2000.02.01 19:56:48 INTERFACE_STATUS.fr 1 changed state to administratively up
2000.02.01 19:56:48 INTERFACE_STATUS.MFR1 changed state to administratively up
2000.02.01 19:56:49 INTERFACE_STATUS.MFR1 changed state to up
2000.02.01 19:56:52 FRAME_RELAY.MULTILINK Link t1 5/1 1 state change: old
state=ACK_RX, new state=UP
2000.02.01 19:56:52 FRAME_RELAY.MFR LIP Link t1 5/1 1 : state changed to UP
2000.02.01 19:56:52 FRAME_RELAY.LMI Port 1 vc 16: VC 16 is inactive because the
port is down.
2000.02.01 19:56:52 FRAME_RELAY.LMI Port 1 vc 16: The switch reports VC 16 as
inactive.
2000.02.01 19:56:52 FRAME_RELAY.LMI fr 1 coming up with last good Rx seq# 63
2000.02.01 19:56:53 FRAME_RELAY.MULTILINK Link t1 5/2 1 state change: old
state=ACK_RX, new state=UP
2000.02.01 19:56:53 FRAME_RELAY.MFR LIP Link t1 5/2 1 : state changed to UP
2000.02.01 19:56:53 INTERFACE_STATUS.fr 1 changed state to up
```

This command shows every state change for the links within the bundle. Notice the first message shows "Link t1 5/1 1" as moving from the old state of IDLE to ADD_SENT. As the negotiation continues, the router will eventually move the link to UP:

```
2000.02.01 19:56:52 FRAME_RELAY.MULTILINK Link t1 5/1 1 state change: old
state=ACK_RX, new state=UP
```

This command can be helpful in monitoring the status of each link without having to fight through LMI messages that can be difficult to follow. Although this command does not give any idea on what is being sent to the other side or how often LMI messages are being sent, it does show if t here is a continued problem with a certain link that continues to change states.

debug frame-relay multilink [hssi <*port*> | t1 <*slot/port*> | t3 <*slot/port*> | e1 <*slot/port*> | dds <*slot/port*>]

Much like "debug frame-relay multilink frame-relay <*port*>," this command will also show the debug information for just part of the network. In this case, the only debug information displayed will consist of whichever port is mentioned.

Common Frame-Relay Problems

In some situations, it may happen that there is a need to use the above debug and show commands above to troubleshoot a particular issue on a link. Below are sample scenarios and the symptoms that cause them followed by a proposed solutions.

No LMI Being Received

The most common problem with Frame Relay is that LMI is not being received from the far end. There are two or three reasons that may cause this to happen. Below are common reasons why a Frame Relay interface may not be receiving LMI:

- LMI Mismatch
- Administratively Down Frame Relay Interface on Far End

The following shows how to check each issue to verify the configuration is correct:

LMI Mismatch

LMI Mismatch can be verified for each side by doing a “show interface frame-relay <*port*>.” Below is an example of the output, the LMI type for this side has been put in bold:

```
fr 1 is DOWN
  Configuration:
    Signaling type is CISCO, signaling role is BOTH
    Multilink disabled
    Polling interval is 10 seconds,
    full inquiry interval is 6 polling intervals
  Link information:
    5 minute input rate 48 bits/sec, 0 packets/sec
    5 minute output rate 48 bits/sec, 0 packets/sec
    BW 1536 Kbps
    Queueing method: weighted fair
    HDLC tx ring limit: 2
    Output queue: 0/1/428/64/0 (size/highest/max total/threshold/drops)
      Conversations  0/1/256 (active/max active/max total)
      Available Bandwidth 1152 kilobits/sec
      358 packets input, 14362 bytes
      1 pkts discarded, 0 error pkts, 0 unknown protocol pkts
      372 packets output, 14892 bytes
      1 tx pkts discarded, 0 tx error pkts
```

To fix this issue, issue the following command from the Frame-Relay interface configuration mode context:
frame-relay lmi-type <*type***>**

Administratively Down Frame-Relay Interface on Far End

After configuration, it is possible that the interface may not have been turned up. To verify the far end has been turned up, do a “show interface” on the port. If the port is administratively down, enter the following command from the Frame-Relay interface configuration mode context:

```
Router(config-fr 1.1)#no shutdown
```

DLCI Stays In An Inactive State

If a DLCI stays in an Inactive state, there is a good chance the other side is either configured for a different DLCI or has not been configured at all. To verify this setup, either of the two following commands can be helpful:

```
show frame-relay pvc
or
show interface frame-relay <number.subinterface number>
```

In an instance where a DLCI number has been misconfigured, the following output may be seen from a "show frame-relay pvc":

```
Frame Relay Virtual Circuit Statistics for interface FR 1
          Active    Inactive   Deleted   Static
  local       0          1          1          2

DLCI = 17, DLCI USAGE = LOCAL, PVC STATUS = DELETED, INTERFACE = FR
1.1
  MTU: 1500
  input pkts:   0       output pkts:  1        in bytes:     0
  out bytes:    198      dropped pkts: 0        in FECN pkts: 0
  in BECN pkts: 0        in DE pkts:   0        out DE pkts:  0
  Creation time: 08-06-2000 15:36:46   Last status change: 00W:00D:00H:00M:01S

DLCI = 16, DLCI USAGE = UNUSED, PVC STATUS = INACTIVE, INTERFACE =
FR 1.2
  MTU: 1500
  input pkts:   0       output pkts:  0        in bytes:     0
  out bytes:    0       dropped pkts: 0        in FECN pkts: 0
  in BECN pkts: 0        in DE pkts:   0        out DE pkts:  0
  Creation time: 08-06-2000 15:36:48   Last status change: 00W:00D:00H:00M:01S
```

This shows that the far side has been set to DLCI 16 since it shows INACTIVE, however the local has been set to DELETED since the frame-relay switch does not have that DLCI created. A simple resolution to this issue would be modifying the Frame-Relay DLCI under the Frame-Relay subinterface configuration mode context:

```
Router(config-fr 1.1)#frame-relay interface-dlci 16
```

DLCI Becomes Active, But Traffic Will Not Pass

In many situations where there is a provider on the far end of the connection, there may be a device configured for a proprietary version of Frame Relay. In this instance, it is possible that the Frame Relay interface may come up and the DLCI will become active. However, no traffic will pass across the interface.. All NetVanta products only support the IETF standard Frame Relay encapsulation. In an instance where the provider is the Far End, it will be necessary to coordinate with them to have their configuration changed to match the IETF standard.

Congestion on Frame Relay Interface

Oftentimes, a Frame Relay interface may be overloaded with traffic. In this instance, BECNs and FECNs may be sent out by the Frame Relay Switch. Any traffic during this instance that is Discard Eligible may be dropped by the Switch. In response to this, a BECN (Backward Explicit Congestion Notification) will be sent towards the transmitting end. These can be monitored through the use of a “show frame-relay pvc”:

```
Frame Relay Virtual Circuit Statistics for interface FR 1
          Active    Inactive   Deleted   Static
  local       1         0          0          1

DLCI = 16, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = FR
1.1
  MTU: 1500
  input pkts:   1       output pkts:  2        in bytes:     30
  out bytes:    228      dropped pkts: 0        in FECN pkts: 0
  in BECN pkts: 0        in DE pkts:   0        out DE pkts:  0
  Creation time: 08-06-2000 15:46:27   Last status change: 00W:00D:00H:00M:06S
```

If the in BECN pkts shows anything but a 0, there has been congestion towards the Frame-Relay switch. In this instance, it may be necessary to configure the CIR and Excess Burst Rate correctly. The commands for configuring these two commands are shown below:

```
Router(config-fr 1.1)#frame-relay bc <CIR Rate>
```

```
Router(config-fr 1.1)#frame-relay be <Excess Rate>
```

More information regarding these two commands can be found above under the section labeled “Configuring Frame Relay Traffic Shaping.