

Configuring Multilink Frame Relay

This chapter describes how to configure Multilink Frame Relay (MFR) interfaces on the ERX system.

Topic	Page
Overview	12-1
References	12-4
Supported MFR Features	12-4
Unsupported MFR Features	12-6
Before You Configure MFR	12-6
Configuration Tasks	12-6
Monitoring MFR	12-9

Overview

MFR aggregates multiple physical links into a single logical bundle. More specifically, MFR bundles multiple link layer channels into a single network layer channel.

The systems joined by the multilink each assign the same unique name to the bundle. A bundle can consist of multiple physical links of the same type—such as multiple asynchronous lines—or can consist of physical links of different types—such as leased synchronous lines and dial-up asynchronous lines.

The system treats MFR like nonmultilink Frame Relay. Packets received with an MFR header are subject to sequencing. Packets received without the MFR header cannot be sequenced and can be delivered only on a first-come, first-served basis.

T1/E1 Connections

Some users need more bandwidth than a T1 or an E1 channel can provide, but cannot afford the expense or do not need the bandwidth of T3 or E3. Equal-cost multipath (ECMP) is one way to achieve a bandwidth greater than DS1 service without going to the expense and infrastructure required for DS3 service. MFR is commonly used as an alternative to ECMP to deliver NxT1 service. Cost-analysis of NxT1 versus DS3 service typically imposes a practical limit of 8xT1 service; that is, aggregation of no more than eight T1 or E1 connections into an MFR bundle.

This implementation of MFR logically aggregates up to eight T1 or E1 connections into a single virtual connection, known as a bundle, to a given customer site. The connections can terminate at a CPE router (see Figure 12-1) or a Multilink Frame Relay bridge (see Figure 12-2).

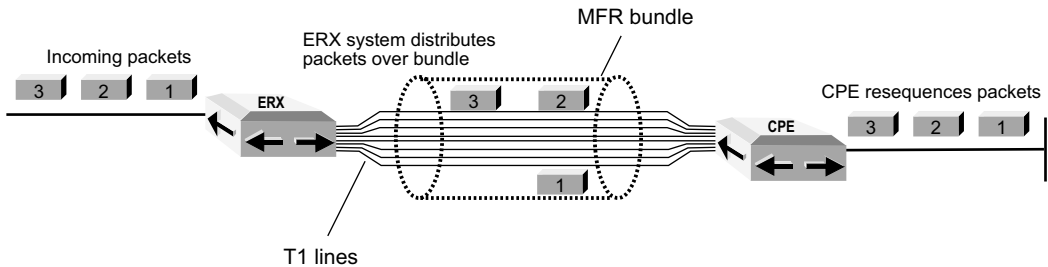


Figure 12-1 MFR aggregation of T1 lines into a single bundle

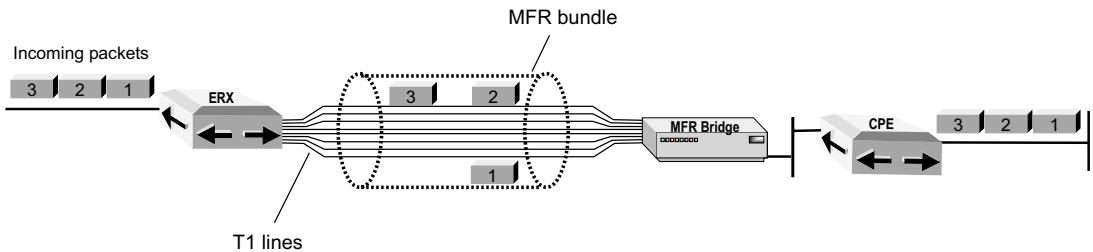


Figure 12-2 Terminating the bundle at an MFR bridge

MFR Link Integrity Protocol

Member links in an MFR bundle support MFR Link Integrity Protocol (LIP). LIP offers several types of messages, which allow member links to join and leave a bundle. See Table 12-1.

Table 12-1 LIP messages and functions

LIP Message	Function
Add-Link	Member link sends this message to request to join a bundle.
Add-Link-Ack	Member link sends this message when it receives an Add-Link message.
Add-Link-Rej	Member link sends this message to reject a request to join a bundle.
Hello	Member link sends this message to check the status of another member.
Hello-Ack	Member link sends this message when it receives a Hello message.
Remove-Link	Member link sends this message to request to leave a bundle.
Remove-Link-Ack	Member link sends this message when it receives a Remove-Link message.

The DTE creates a link management interface (LMI) with the network by encapsulating the Frame Relay frame within an MFR frame. You assign one or more data link control identifiers (DLCIs) to a bundle.

Interface Stacking

Because MFR aggregates multiple link layer channels onto a single network layer IP interface, protocol layering within the system is different than it is for nonmultilink Frame Relay. See Figure 12-3.

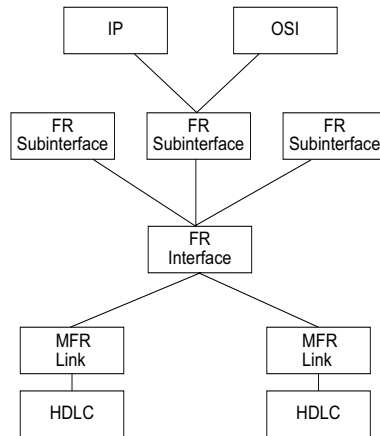


Figure 12-3 Structure of MFR

MFR Link Integrity Protocol runs on each link in a bundle. However, from the major Frame Relay interface (the bundle) upward, the interface stacking is the same as for nonmultilink Frame Relay. For example, LMI runs only on the bundle. The bundle sends and receives all MFR packets.

References

For more information about the MFR protocol, consult the following resources:

- Multilink Frame Relay UNI/NNI Implementation Agreement, FRF.16 (April 2000)
- Frame Relay Fragmentation Implementation Agreement, FRF.12 (December 1997)

Supported MFR Features

The system supports the following MFR features on the CE1, cOCx/STMx, CT1, and CT3 line modules:

- Logical aggregation of up to eight T1 or E1 links in a bundle
 - Monotonically increasing sequence numbers for each circuit
- All packets distributed across the member links have monotonically increasing sequence numbers for each circuit. This feature enables the remote system on the customer premises to perform resequencing (if it is configured to do so).
- Static configuration of the links participating in a multilink bundle
 - Round-robin packet distribution
 - > On CE1, CT1, and CT3 line modules, packet distribution across the member links in a bundle is handled only in a round-robin fashion. The round-robin approach is used even when the member links have different line rates.
 - > On cOCx/STMx line modules, the system keeps track of the link with the least traffic. If this link cannot forward a packet, the system attempts to forward the traffic on a different link. If this attempt also fails, the system uses a round-robin approach.

You can configure bundles as follows:

- On a CE1 line module and its corresponding I/O module, you can configure:
 - > Member links from different CE1 ports in the same bundle
 - > Any combination of bundles that does not exceed the 20 available E1 channels (for example, 20 single-link E1 bundles or 2 eight-link bundles and 4 single-link bundles)
- On a cOCx/STMX line module and its corresponding I/O module, you can configure:
 - > Member links from different OC3/STM1 ports in the same bundle
 - > Any combination of bundles that does not exceed the 336 available T1 channels (for example, 336 single-link T1 bundles, 42 eight-link bundles, or 41 eight-link bundles and 8 single-link bundles)
 - > Any combination of bundles that does not exceed the 252 available E1 channels (for example, 252 single-link T1 bundles, 34 eight-link bundles, or 33 eight-link bundles and 8 single-link bundles)
- On a CT1 line module and its corresponding I/O module, you can configure:
 - > Member links from different CT1 ports in the same bundle
 - > Any combination of bundles that does not exceed the 24 available T1 channels (for example, 24 single-link T1 bundles, 3 eight-link bundles, or 2 eight-link bundles and 8 single-link bundles.)
- On a CT3 or CT3/T3 FO line module and its corresponding I/O module, you can configure:
 - > Only member links from the same T3 interfaces into the same bundle. You cannot configure member links from different T3 ports in the same bundle.
 - > Any combination of bundles that does not exceed the 28 available T1 channels per port (for example, 28 single-link T1 bundles or 3 eight-link bundles and 4 single-link bundles per port)

Unsupported MFR Features

The system does not support the following MFR features:

- Fragmentation

The system does not support MFR fragmentation or reassembly. When using MFR on the system, configure all peer devices so that they do not fragment MFR frames. The system drops all fragmented frames that it receives.

- Resequencing of out-of-order packets in the absence of fragmentation

Given the location in the network where the system resides, the NxT1 links to a customer site represent one of many places across the IP network where packets might be received out of order. For example, if the system has multiple uplinks to a core router, packets might be received out of order across these links. Packet resequencing is therefore left as an exercise for the end station rather than the aggregation router.

Before You Configure MFR

Before you begin configuring MFR, you must configure the physical interfaces that will be aggregated by MFR.

The procedures described in this chapter assume that a physical interface, such as a T1 or T3 interface, has been configured.

Configuration Tasks

MFR configuration consists of three major tasks, each with several steps:

- 1 Create the member links to be aggregated into a multilink bundle.

- a Specify the interface on which you want to configure MFR.

```
host1(config)#interface serial 2/0:1
```

- b Specify MFR as the encapsulation method on the interface.

```
host1(config-if)#encapsulation mframe-relay ietf
```

- 2 Add member links to a multilink bundle.

- a Define the MFR bundle.

```
host1(config)#interface mframe-relay boston
```

- b Add each member link.

```
host1(config-if)#member-interface serial 2/0:1
```

- c (Optional) Add a description to the major interface.

```
host1(config-if)#frame-relay description  
    bostonBundleDescription
```

- d (Optional) Configure Frame Relay parameters.

```
host1(config-if)#frame-relay intf-type dce  
host1(config-if)#frame-relay lmi-type cisco
```

3 Configure the Frame Relay subinterface.

- a Define the subinterface for the MFR bundle.

```
host1(config)#interface mlframe-relay boston.1
```

- b Assign a DLCI for the subinterface.

```
host1(config-subif)#frame-relay interface-dlci 16 ietf
```

- c (Optional) Add a description to the subinterface.

```
host1(config-subif)#frame-relay description  
    bostonBundleSubOneDescription
```

- d Assign an IP address to the subinterface.

```
host1(config-subif)#ip address 10.10.100.1 255.255.255.0
```

Example The following example configures three T1 lines and aggregates them into a multilink bundle called *boston*.

```
host1(config)#interface serial 2/0:1  
host1(config-if)#encapsulation mlframe-relay ietf  
host1(config-if)#exit  
host1(config)#interface serial 2/0:2  
host1(config-if)#encapsulation mlframe-relay ietf  
host1(config-if)#exit  
host1(config)#interface serial 2/0:3  
host1(config-if)#encapsulation mlframe-relay ietf  
host1(config-if)#exit  
host1(config)#interface mlframe-relay boston  
host1(config-if)#member-interface serial 2/0:1  
host1(config-if)#frame-relay description  
    bostonBundleDescription  
host1(config-if)#frame-relay intf-type dce  
host1(config-if)#frame-relay lmi-type cisco  
host1(config-if)#member-interface serial 2/0:2  
host1(config-if)#member-interface serial 2/0:3  
host1(config-if)#exit  
host1(config)#interface mlframe-relay boston.1
```

```
host1(config-subif)frame-relay description
    bostonBundleSubOneDescription
host1(config-subif)#frame-relay interface-dlci 16 ietf
host1(config-subif)#ip address 10.10.100.1 255.255.255.0
```

Configuring Frame Relay Versus MFR

All the configuration commands that apply to Frame Relay also apply to MFR. The following listing describes commands specific to configuring MFR; for other Frame Relay commands, see *Chapter 11, Configuring Frame Relay*.

encapsulation ietf

- Use to configure MFR as the encapsulation method on an individual interface.
- Use this command only within the context of an individual interface. Issuing this command creates an MFR link, also referred to as an MFR bundle member.
- Example

```
host1(config)#interface serial 2/0:1
host1(config-if)#encapsulation mlframe-relay ietf
```
- Use the **no** version to disable MFR on an interface.

frame-relay description

- Use to assign a text description or an alias to a MFR major interface or subinterface.
- Description name can be up to 64 characters.
- Use the **no** version to remove the text description or alias from the MFR major interface or subinterface.

interface mlframe-relay

- Use to create a Frame Relay major interface, also known as the MFR bundle.
- Example

```
host1(config-if)#interface mlframe-relay group2
```
- Use the **no** version to delete the MFR bundle.

member-interface

- Use to add an MFR interface—also known as an MFR bundle member—to an MFR bundle.
- Example

```
host1(config-if)#member-interface serial 2/0:1
```
- Use the **no** version to remove the specified interface from the MFR bundle.

Monitoring MFR

Use the commands in this section to display information about MFR interfaces.

You can set a statistics baseline for an MFR link with the **baseline frame-relay multilink interface** command. Similarly, you can set a statistics baseline for an MFR bundle using the **baseline frame-relay interface mlframe-relay** command. Use the **delta** keyword with the **show** commands described below to display statistics with the baseline values subtracted.

After you configure multilink Frame Relay, you can use the **show frame-relay** commands to view information about the multilink. For information about these commands, see *Chapter 11, Configuring Frame Relay*.

You can use the output filtering feature of the **show** command to include or exclude lines of output based on a text string you specify. Refer to *show Commands in ERX System Basics Configuration Guide, Chapter 2, Command Line Interface*, for details.

baseline frame-relay interface

- Use to set a statistics baseline for MFR bundles, Frame Relay interfaces, subinterfaces, and circuits.
- The system implements the baseline by reading and storing the statistics at the time the baseline is set and then subtracting this baseline whenever baseline-relative statistics are retrieved.
- You cannot set a baseline for groups of interfaces, subinterfaces, or circuits. You must set baselines one at a time.
- When baselining is requested, the time since the last baseline was set is displayed in *hours:minutes:seconds* or *days/hours* format. If a baseline has not been set, the message “No baseline has been set” is displayed instead.
- The regular interface statistics and LMI statistics for interfaces are subject to the same baseline timestamp. You cannot set separate baselines for these statistics.
- Use the optional **delta** keyword with Frame Relay **show** commands to specify that baselined statistics are to be shown.
- Example

```
host1#baseline frame-relay interface mlframe-relay
```
- There is no **no** version.

baseline frame-relay multilinkinterface

- Use to set a statistics baseline for MFR links.
- The system implements the baseline by reading and storing the statistics at the time the baseline is set and then subtracting this baseline whenever baseline-relative statistics are retrieved.
- When baselining is requested, the time since the last baseline was set is displayed in *hours:minutes:seconds* or *days/hours* format. If a baseline has not been set, the message “No baseline has been set” is displayed instead.
- The regular interface statistics and LIP statistics for interfaces are subject to the same baseline timestamp. You cannot set separate baselines for these statistics.
- Use the optional **delta** keyword with Frame Relay **show** commands to specify that baselined statistics are to be shown.
- Example


```
host1#baseline frame-relay multilinkinterface
```
- There is no **no** version.

show frame-relay interface

- Use to display the information about the Frame Relay layer of the interface.
- Use the optional **delta** keyword to specify that baselined statistics are to be shown.
- Use the **brief** keyword to display the operational status of all configured interfaces.
- Field descriptions
 - › Frame relay interface mlframe-relay – name of the MFR bundle
 - › Status of the major Frame Relay interface – one of the following states:
 - Up – traffic can flow on the interface
 - Offline – traffic cannot flow because hardware is unavailable
 - Down – traffic cannot flow because of a problem in the interface at the current protocol layer
 - LowerLayerDown – traffic cannot flow because of a problem in an interface at a lower protocol layer
 - AdministrativelyDown – traffic cannot flow because of manual administrative intervention
 - › Number of interface down transitions – number of interfaces that have changed to a down state
 - › Time since last status change – time since the interface last changed its state
 - › In bytes – number of inbound bytes received on the interface
 - › In frames – number of inbound frames received on the interface
 - › In errors – number of inbound errors received on the interface
 - › In discards – number of inbound packets discarded
 - › In unknown protos – number of packets received on the interface with unknown protocols

- › Out bytes – number of outbound bytes transmitted on the interface
- › Out frames – number of outbound frames transmitted on the interface
- › Out errors – number of outbound errors transmitted on the interface
- › Out discards – number of outbound packets discarded

- Example 1

```
host1#show frame-relay interface brief
Frame relay interface mlframe-relayTEST, status is up
```

- Example 2

```
host1#show frame-relay interface mlframe-relay TEST
Frame relay interface mlframe-relayTEST, status is up
Number of interface down transitions is 0
Time since last status change 00:01:47
Number of configured circuits: 2
  In bytes: 452           Out bytes: 198
  In frames: 19          Out frames: 11
  In errors: 0           Out errors: 0
  In discards: 8         Out discards: 0
  In unknown protos: 0
```

- Example 3

```
host1#show frame-relay interface mlframe-relay members
Frame relay interface mlframe-relay TEST is up
  Frame relay multilink member-interface 4/0:1 is up
  Frame relay multilink member-interface 4/1:1 is up
```

show frame-relay lip

- Use to display the state of MFR LIP on an MFR link.
- Use the **delta** keyword to specify that baselined statistics are to be shown.
- Use the **brief** keyword to display the operational status of all configured interfaces.
- Field descriptions
 - › Frame relay interface – specifier for the Frame Relay interface
 - › Status of the major Frame Relay interface – one of the following states:
 - Up – traffic can flow on the interface
 - Offline – traffic cannot flow because hardware is unavailable
 - Down – traffic cannot flow because of a problem in the interface at the current protocol layer
 - LowerLayerDown – traffic cannot flow because of a problem in an interface at a lower protocol layer
 - AdministrativelyDown – traffic cannot flow because of manual administrative intervention
 - › Number of interface down transitions – number of interfaces that have changed to a down state

- › Time since last status change – time since the interface last changed its state
- › Add Links sent – number of Add Link messages sent from this interface
- › Add Links received – number of Add Link messages received on this interface
- › Add Link Acknowledgments sent – number of Add Link acknowledgments sent from this interface
- › Add Link Acknowledgments received – number of Add Link acknowledgments received on this interface
- › Add Link Rejects sent – number of Add Link Reject messages sent from this interface
- › Add Link Rejects received – number of Add Link Reject messages received on this interface
- › Hellos sent – number of Hello messages sent from this interface
- › Hellos received – number of Hello messages received on this interface
- › Hello Acknowledgments sent – number of Hello messages sent from this interface
- › Hello Acknowledgments received – number of Hello messages received on this interface
- › Remove Links sent – number of Remove Link messages sent from this interface
- › Remove Links received – number of Remove Link messages received on this interface
- › Remove Link Acknowledgments sent – number of Remove Link acknowledgments sent from this interface
- › Remove Link Acknowledgments received – number of Remove Link acknowledgments received on this interface

- Example 1

```
host1#show frame-relay lip brief
LIP information for frame relay interface 4/0:1, status is
  up
Number of interface down transitions is 0
Time since last status change 00:03:16

LIP information for frame relay interface 4/1:1, status is
  up
Number of interface down transitions is 0
Time since last status change 00:03:20
```

- Example 2

```
host1#show frame-relay lip interface serial 4/0:1
LIP information for frame relay interface 4/0:1, status is
  up
Number of interface down transitions is 0
Time since last status change 00:05:19
  Add Links sent: 1
  Add Links received: 1
```

```
Add Link Acknowledgements sent: 1
Add Link Acknowledgements received: 1
Add Link Rejects sent: 0
Add Link Rejects received: 0
Hellos sent: 32
Hellos received: 31
Hello Acknowledgements sent: 31
Hello Acknowledgements received: 32
Remove Links sent: 0
Remove Links received: 0
Remove Link Acknowledgements sent: 0
Remove Link Acknowledgements received: 0
```

show frame-relay lmi

- Use to display configuration and state information and statistics about the LMI.
- You can specify an interface type and location.
- Use the **brief** keyword to display abbreviated PVC information.
- Use the **delta** keyword to specify that baselined statistics are to be shown.
- DTE field descriptions
 - › Frame relay DTE interface mlframe-relay – name of the MFR bundle
 - › N391 – value of N391 full-status polling counter
 - › N392 – value of N392 error threshold counter
 - › N393 – value of N393 monitored events counter
 - › T391 – value of T391 link integrity polling timer interval
 - › Configured LMI type: one of the following options:
 - ANSI – ANSI-T1.617 Annex D
 - Q933A – ITU-T Q.933 Annex A
 - Cisco – original *Group of Four* specification developed by DEC, Northern Telecom, Stratacom, and Cisco
 - None – suppresses LMI
 - › status is up – availability of MFR bundle: up or down
 - › Number of interface down transitions – number of times the interface has become unavailable
 - › Time since last status change – elapsed time since LMI information changed
 - Enquiries sent – total number of LMI status inquiries sent by the DTE on this interface
 - Full enquiries sent – total number of LMI full status inquiries sent by the DTE on this interface
 - Enquiry responses received – total number of LMI full and regular status responses received by the DTE on this interface
 - Full enquiry responses received – total number of LMI full status responses received by the DTE on this interface
 - Async updates received – total number of asynchronous LMI updates received by the DTE on this interface

- Unknown messages received – total number of unknown LMI messages received on this interface
- Loss of sequencing detected – total number of times a loss of sequencing in received LMI messages was detected by the DTE on this interface
- No response timeouts – total number of times a timeout occurred without receiving a response to an LMI request by the DTE on this interface
- Last sequence number sent – last sequence number sent on this interface
- Last sequence number received – last sequence number received on this interface
- DCE field descriptions:
 - › Frame relay DCE interface mframe-relay – name of the MFR bundle
 - › N391 – value of N391 full-status polling counter
 - › N392 – value of N392 error threshold counter
 - › T392 – value of T392 polling verification timer
 - › Configured LMI type: one of the following options:
 - ANSI – ANSI-T1.617 Annex D
 - Q933A – ITU-T Q.933 Annex A
 - Cisco – original *Group of Four* specification developed by DEC, Northern Telecom, Stratacom, and Cisco
 - None – suppresses LMI
 - › status is up – availability of MFR bundle: up or down
 - › Number of interface down transitions – number of times the interface has become unavailable
 - › Time since last status change – elapsed time since LMI information changed
 - Enquiries received – total number of LMI status inquiries received by the DCE on this interface
 - Enquiry responses sent – total number of LMI status responses sent by the DCE on this interface
 - Full enquiry responses sent – total number of LMI full status responses sent by the DCE on this interface
 - Async updates sent – total number of LMI ASYNC updates sent by the DCE on this interface
 - Unknown messages received – total number of unknown LMI messages received on this interface
 - Loss of sequencing detected – total number of times a loss of sequencing in received LMI messages was detected by the DCE on this interface
 - No response timeouts – total number of times a timeout occurred without receiving a status inquiry from the DTE on this interface
 - Last sequence number sent – last sequence number sent on this interface
 - Last sequence number received – last sequence number received on this interface

- Example 1

```
host1#show frame-relay lmi brief
LMI information for frame relay DTE interface
mlframe-relayTEST
DTE parameter N391 is 6, N392 is 3, N393 is 4, T391 is 10
Configured LMI type is ANSI, status is up
Number of interface down transitions is 0
Time since last status change 00:05:39
```

- Example 2

```
host1#show frame-relay lmi interface mlframe-relay TEST
LMI information for frame relay DTE interface
mlframe-relayTEST
DTE parameter N391 is 6, N392 is 3, N393 is 4, T391 is 10
Configured LMI type is ANSI, status is up
Number of interface down transitions is 0
Time since last status change 00:06:20
  Enquiries sent: 39
  Full enquiries sent: 7
  Enquiry responses received: 39
  Full enquiry responses received: 7
  Async updates received: 0
  Unknown messages received: 0
  Loss of sequencing detected: 0
  No response timeouts: 0
  Last sequence number sent: 39
  Last sequence number received: 39
```

show frame-relay map

- Use to display the current Frame Relay and MFR map entries.
- Field descriptions
 - › subinterface – name and subinterface number of the MFR bundle in the format *bundle-name.subinterface-number*
 - › State of the subinterface – one of the following states:
 - Up – traffic can flow on the interface
 - Offline – traffic cannot flow because hardware is unavailable
 - Down – traffic cannot flow because of a problem in the interface at the current protocol layer
 - LowerLayerDown – traffic cannot flow because of a problem in an interface at a lower protocol layer
 - AdministrativelyDown – traffic cannot flow because of manual administrative intervention
 - › DLCI number – decimal value, hexadecimal value, and value as it appears on the wire of DLCI

- Example

```
host1#show frame-relay map
Frame relay sub-interface mlframe-relayTEST.1 (up): DLCI
 16(0x10,0x4)
Frame relay sub-interface mlframe-relayTEST.2 (up): DLCI
 17(0x11,0x14)
```

show frame-relay multilinkInterface

- Use to display the statistics about all MFR interfaces or the specified MFR interfaces.
- Field descriptions
 - › Multilink Frame relay interface – specifier for the Frame Relay interface
 - › State of the MFR interface – one of the following states:
 - Up – traffic can flow on the interface
 - Offline – traffic cannot flow because hardware is unavailable
 - Down – traffic cannot flow because of a problem in the interface at the current protocol layer
 - LowerLayerDown – traffic cannot flow because of a problem in an interface at a lower protocol layer
 - AdministrativelyDown – traffic cannot flow because of manual administrative intervention
 - › Number of multilink interface down transitions – number of interfaces that have changed to a down state
 - › Time since last status change – time since the interface last changed its state
 - › In bytes – number of inbound bytes received on the interface
 - › In frames – number of inbound frames received on the interface
 - › In errors – number of inbound errors received on the interface
 - › In discards – number of inbound packets discarded
 - › In unknown protos – number of packets received on the interface with unknown protocols
 - › Out bytes – number of outbound bytes transmitted on the interface
 - › Out frames – number of outbound frames transmitted on the interface
 - › Out errors – number of outbound errors transmitted on the interface
 - › Out discards – number of outbound packets discarded
- Example

```
Multilink Frame relay interface 6/2:2, status is down
Number of multilink interface down transitions is 0
Time since last status change 2 days, 23 hours
  In bytes: 0                Out bytes: 0
  In frames: 0              Out frames: 0
  In errors: 0              Out errors: 0
  In discards: 0            Out discards: 0
  In unknown protos: 0
```

show frame-relay pvc

- Use to display statistics about PVCs for Frame Relay interfaces.
- Specify a DLCI number or an interface type and location.
- Use the optional **delta** keyword to specify that baselined statistics are to be shown.
- The **brief** keyword displays abbreviated PVC information.
- Field descriptions
 - › DLCI – DLCI number
 - › subinterface – name and subinterface number of the MFR bundle in the format *bundle-name.subinterface-number*
 - › status – status of the PVC
 - › Number of circuit status inactive transitions – number of times a circuit came down because of error conditions
 - › Time since creation – time since PVC was created
 - › last status change – time since PVC status last changed
 - › In pkts – number of incoming packets received on the circuit
 - › Out pkts – number of outgoing packets transmitted on the circuit
 - › In bytes – number of input bytes received on the circuit
 - › Out bytes – number of output bytes received on the circuit
 - › In FECN pkts – number of packets received with the forward explicit congestion notification (FECN) bit set. The FECN bit is set by a network to notify the user that data traffic may experience congestion in the direction of the frame carrying the FECN bit. The FECN bit is set by the network (not by the transmitting user), and there is no obligation for end systems to take any action regarding the FECN bit.
 - › Out FECN pkts – number of packets transmitted with the FECN bit set
 - › In BECN pkts – number of packets received with the backward explicit congestion notification (BECN) bit set. The BECN bit is set by a network to notify the user that data traffic may experience congestion in the opposite direction of the frame carrying the BECN bit. The BECN bit is set by the network, and there is no obligation for end systems to take any action regarding the BECN bit.
 - › Out BECN pkts – number of packets transmitted with the BECN bit set
 - › In DE pkts – number of packets received with the discard eligibility (DE) bit set. When the DE bit is set, it indicates that the frame should be discarded in preference to other frames without the DE bit set. The DE bit may be set by the network or the user. Once it is set, it cannot be reset by the user.
 - › Out DE pkts – number of packets transmitted with the DE bit set
 - › Dropped packets – number of dropped packets

- Example 1

```
host1#show frame-relay pvc brief
```

```
PVC information for frame relay DTE interface  
mlframe-relayTEST
```

```
DLCI 16 in sub-interface mlframe-relayTEST.1, status is  
active
```

```
DLCI 17 in sub-interface mlframe-relayTEST.2, status is  
active
```

- Example 2

```
host1#show frame-relay pvc interface mlframe-relay TEST
```

```
PVC information for frame relay DTE interface  
mlframe-relayTEST
```

```
DLCI 16 in sub-interface mlframe-relayTEST.1, status is  
active
```

```
Number of circuit status inactive transitions is 0
```

```
Time since creation 00:07:20, last status change 00:07:11
```

```
In pkts: 14           Out pkts: 0  
In bytes: 420        Out bytes: 0  
In FECN pkts: 0     Out FECN pkts: 0  
In BECN pkts: 0     Out BECN pkts: 0  
In DE pkts: 0       Out DE pkts: 0  
Dropped pkts: 14
```

```
DLCI 17 in sub-interface mlframe-relayTEST.2, status is  
active
```

```
Number of circuit status inactive transitions is 0
```

```
Time since creation 00:07:20, last status change 00:07:11
```

```
In pkts: 14           Out pkts: 0  
In bytes: 420        Out bytes: 0  
In FECN pkts: 0     Out FECN pkts: 0  
In BECN pkts: 0     Out BECN pkts: 0  
In DE pkts: 0       Out DE pkts: 0  
Dropped pkts: 14
```

show frame-relay subinterface

- Use to display the state of the subinterface.
- The subinterface can be in one of the following states:
 - › Up – traffic can flow on the interface
 - › Offline – traffic cannot flow because hardware is unavailable
 - › Down – traffic cannot flow because of a problem in the interface at the current protocol layer

- › LowerLayerDown – traffic cannot flow because of a problem in an interface at a lower protocol layer
- › AdministrativelyDown – traffic cannot flow because of manual administrative intervention
- The **brief** keyword displays only the operational status of all configured subinterfaces.
- Use the optional **delta** keyword to specify that baselined statistics are to be shown.
- Field descriptions
 - › Frame relay sub-interface `mframe-relay` – name and subinterface number of the MFR bundle in the format *bundle-name.subinterface-number*
 - › status – state of the subinterface, as follows:
 - Up – traffic can flow on the interface
 - Offline – traffic cannot flow because hardware is unavailable
 - Down – traffic cannot flow because of a problem in the interface at the current protocol layer
 - LowerLayerDown – traffic cannot flow because of a problem in an interface at a lower protocol layer
 - AdministrativelyDown – traffic cannot flow because of manual administrative intervention
 - › Number of sub-interface down transitions – number of times a subinterface came down because of error conditions
 - › Time since last status change – time since the last status change on the subinterface
 - › In bytes – number of inbound bytes received on the subinterface
 - › Out bytes – number of outbound bytes transmitted on the subinterface
 - › In frames – number of inbound frames received on the interface
 - › Out frames – number of outbound frames transmitted on the interface
 - › In errors – number of inbound errors received on the subinterface
 - › Out errors – number of outbound errors transmitted on the subinterface
 - › In discards – number of inbound packets discarded
 - › Out discards – number of outbound packets discarded
 - › In unknown protos – number of packets received on the subinterface with unknown protocols

- Example 1

```
host1#show frame-relay subinterface brief
Frame relay sub-interface mframe-relayTEST.1, status is up
Frame relay sub-interface mframe-relayTEST.2, status is up
```

- Example 2

```
host1#show frame-relay subinterface mframe-relay TEST
Frame relay sub-interface mframe-relayTEST.1, status is up
Number of sub-interface down transitions is 0
```

```
Time since last status change 00:07:49
  In bytes: 512                Out bytes: 0
  In frames: 16               Out frames: 0
  In errors: 0                Out errors: 0
  In discards: 16            Out discards: 0
  In unknown protos: 0
```

```
Frame relay sub-interface mlframe-relayTEST.2, status is up
Number of sub-interface down transitions is 0
Time since last status change 00:07:50
  In bytes: 512                Out bytes: 0
  In frames: 16               Out frames: 0
  In errors: 0                Out errors: 0
  In discards: 16            Out discards: 0
  In unknown protos: 0
```

show frame-relay summary

- Use to scan all defined Frame Relay interfaces and circuits and to report the status for each discovered interface and circuit as follows:
 - › Up – traffic can flow on the interface
 - › Down – traffic cannot flow because of a problem in the network
 - › Unavailable – traffic cannot flow because hardware is unavailable
- Example

```
host1#show frame-relay summary
2 multilink interface(s) defined, 2 up, 0 down
1 interface(s) defined, 1 up, 0 down
2 sub-interface(s) defined, 2 up, 0 down
2 circuit(s) defined, 2 up, 0 down
```