

# debug dmsp fax-to-doc

To display debug messages for doc MSP fax-to-doc, use the **debug dmsp fax-to-doc** command in privileged EXEC mode. To disable the debug messages, use the **no** form of this command.

**debug dmsp fax-to-doc [tiff-writer]**

**no debug dmsp fax-to-doc [tiff-writer]**

<b>Syntax Description</b>	<b>tiff-writer</b>	(Optional) Displays debug messages that occur while the DocMSP Component is receiving T4 fax data and producing TIFF packets.
---------------------------	--------------------	---

<b>Defaults</b>	No default behavior or values.
-----------------	--------------------------------

<b>Command Modes</b>	Privileged EXEC
----------------------	-----------------

<b>Command History</b>	<b>Release</b>	<b>Modification</b>
	12.1(3)XI	This command was introduced on the Cisco AS5300 access server.

**Examples** The following example displays output from the **debug dmsp fax-to-doc** command.

```
Router# debug dmsp fax-to-doc

*Oct 16 08:29:54.487: docmsp_call_setup_request: callid=22
*Oct 16 08:29:54.487: docmsp_call_setup_request(): ramp data dir=OFFRAMP, conf dir=SRC
*Oct 16 08:29:54.487: docmsp_caps_ind: call id=22, src=21
*Oct 16 08:29:54.487: docmsp_bridge cfid=15, srccid=22, dstcid=21

*Oct 16 08:29:54.487: docmsp_bridge(): ramp data dir=OFFRAMP, conf dir=SRC, encode out=2
*Oct 16 08:29:54.487: docmsp_bridge cfid=16, srccid=22, dstcid=17

*Oct 16 08:29:54.487: docmsp_bridge(): ramp data dir=OFFRAMP, conf dir=DEST, encode out=2
*Oct 16 08:29:54.487: docmsp_xmit: call id src=17, dst=22
*Oct 16 08:29:54.487: docmsp_process_rcv_data: call id src=17, dst=22
*Oct 16 08:29:54.487: offramp_data_process:
*Oct 16 08:29:54.515: docmsp_get_msp_event_buffer:
*Oct 16 08:29:56.115: docmsp_call_setup_request: callid=24
*Oct 16 08:29:56.115: docmsp_call_setup_request(): ramp data dir=ONRAMP, conf dir=DEST
*Oct 16 08:29:56.115: docmsp_caps_ind: call id=24, src=20
*Oct 16 08:29:56.115: docmsp_bridge cfid=17, srccid=24, dstcid=20
```

<b>Related Commands</b>	<b>Command</b>	<b>Description</b>
	<b>debug dmsp doc-to-fax</b>	Displays debug messages for the doc Media Service Provider TIFF or text2Fax engine.

# debug drip event

To display debug messages for Duplicate Ring Protocol (DRiP) events, use the **debug drip event** command in privileged EXEC mode. Use the **no** form of this command to disable debugging output.

**debug drip event**

**no debug drip event**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debugging is disabled for DRiP events.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3(4)T	This command was introduced.

**Usage Guidelines** When a TrBRF interface is configured on the RSM, the DRiP protocol is activated. The DRiP protocol adds the VLAN ID specified in the router command to its database and recognizes the VLAN as a locally configured, active VLAN.

**Examples** The following examples show output for the **debug drip event** command.

DRiP gets a packet from the network:

```
612B92C0: 01000C00 00000000 0C501900 0000AAAA .....P....**
612B92D0: 0300000C 00020000 00000100 0CCCCCCC .....LLL
612B92E0: 00000C50 19000020 AAAA0300 000C0102 ...P... **.....
612B92F0: 01010114 00000002 00000002 00000C50 .....P
612B9300: 19000001 04C00064 04 .....@.d.
```

DRiP gets a packet from the network:

```
Recvd. pak
```

DRiP recognizes that the VLAN ID it is getting is a new one from the network:

```
6116C840:                               0100 0CCCCCCC          ...LLL
6116C850: 00102F72 CBF00024 AAAA0300 000C0102 ../rK{.$**.....
6116C860: 01FF0214 0002E254 00015003 00102F72 .....bT..P../r
6116C870: C8000010 04C00014 044003EB 14          H....@...@.k.
DRIP : remote update - Never heard of this vlan
```

DRiP attempts to resolve any conflicts when it discovers a new VLAN. The value action = 1 means to notify the local platform of change in state.

```
DRIP : resolve remote for vlan 20 in VLAN0
DRIP : resolve remote - action = 1
```

The local platform is notified of change in state:

```
DRIP Change notification active vlan 20
Another new VLAN ID was received in the packet:
DRIP : resolve remote for vlan 1003 in Vlan0
```

No action is required:

```
DRIP : resolve remote - action = 0
```

Thirty seconds have expired, and DRiP sends its local database entries to all its trunk ports:

```
DRIP : local timer expired
DRIP : transmit on 0000.0c50.1900, length = 24
612B92C0: 01000C00 00000000 0C501900 0000AAAA .....P....**
612B92D0: 0300000C 00020000 00000100 0CCCCCCC .....LLL
612B92E0: 00000C50 19000020 AAAA0300 000C0102 ...P... **.....
612B92F0: 01FF0114 00000003 00000002 00000C50 .....P
612B9300: 19000001 04C00064 04 .....@.d.
```

# debug drip packet

To display debug messages for DRiP packets, use the **debug drip packet** command in privileged EXEC mode. Use the **no** form of this command to disable debugging output.

**debug drip packet**

**no debug drip packet**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debugging is not enabled for DRiP packets.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3(4)T	This command was introduced.

**Usage Guidelines** Before you use this command, you can optionally use the **clear drip** command first. As a result the DRiP counters are reset to 0. If the DRiP counters begin to increment, the router is receiving packets.

**Examples** Following is sample output for the **debug drip packet** command.

The following type of output is displayed when a packet is entering the router and you use the **show debug** command:

```
039E5FC0:      0100 0CCCCCCC 00E0A39B 3FFB0028      ...LLL.~#.?.{.(
039E5FD0:  AAAA0300 000C0102 01FF0314 0000A5F6  **.....%v
039E5FE0: 00008805 00E0A39B 3C000000 04C00028  ....~#.<....@.(
039E5FF0: 04C00032 044003EB 0F          .@.2.@.k.
039FBD20:                01000C00 00000010      .....
```

The following type of output is displayed when a packet is sent by the router:

```
039FBD30: A6AEB450 0000AAAA 0300000C 00020000  &.4P...**.....
039FBD40: 00000100 0CCCCCCC 0010A6AE B4500020  ....LLL..&.4P.
039FBD50: AAAA0300 000C0102 01FF0114 00000003  **.....
039FBD60: 00000002 0010A6AE B4500001 04C00064  ....&.4P...@.d
039FBD70: 04          .
```

Related Commands	Command	Description
	<a href="#">debug drip event</a>	Displays debug messages for DRIP events.

# debug dsc clock

To display debugging output for the time-division multiplexing (TDM) clock-switching events on the dial shelf controller (DSC), use the **debug dsc clock** command in privileged EXEC mode. To turn off debugging output, use the **no** form of this command.

[execute-on] **debug dsc clock**

[execute-on] **no debug dsc clock**

**Syntax Description** This command has no arguments or keywords; however, it can be used with the **execute-on** command.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3(2)AA	This command was introduced.

**Usage Guidelines** To perform this command from the router shelf on the Cisco AS5800 series platform, use the **execute-on slot slot-number debug dsc clock** form of this command.

The **debug dsc clock** command displays TDM clock-switching events on the dial shelf controller. The information displayed includes the following:

- Clock configuration messages received from trunks via NBUS
- Dial shelf controller clock configuration messages from the router shelf over the dial shelf interface link
- Clock switchover algorithm events

**Examples** The following example shows that the **debug dsc clock** command has been enabled, and that trunk messages are received, and that the configuration message has been received:

```
AS5800# debug dsc clock
```

```
Dial Shelf Controller Clock debugging is on
AS5800#
00:02:55: Clock Addition msg of len 12 priority 8 from slot 1 port 1 on line 0
00:02:55: Trunk 1 has reloaded
```

Related Commands	Command	Description
	<b>execute-on</b>	Executes commands remotely on a line card.
	<b>show dsc clock</b>	Displays information about the dial shelf controller clock.

# debug dsip

To display debugging output for Distributed System Interconnect Protocol (DSIP) used between a router shelf and a dial shelf, use the **debug dsip** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug dsip {all | api | boot | console | trace | transport}
```

```
no debug dsip {all | api | boot | console | trace | transport}
```

## Syntax Description

<b>all</b>	View all DSIP debugging messages.
<b>api</b>	View DSIP client interface (API) debugging messages.
<b>boot</b>	View DSIP booting messages that are generated when a download of the feature board image is occurring properly.
<b>console</b>	View DSIP console operation while debugging.
<b>trace</b>	Enable logging of header information concerning DSIP packets entering the system into a trace buffer. This logged information can be viewed with the <b>show dsip tracing</b> command.
<b>transport</b>	Debug the DSIP transport layer, the module that interacts with the underlying physical media driver.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
11.3(2)AA	This command was introduced.

## Usage Guidelines

The **debug dsip** command is used to enable the display of debugging messages for DSIP between the router shelf and the dial shelf. Using this command, you can display booting messages generated when the download of an image occurs, view console operation, and trace logging of MAC header information and DSIP transport layer information as modules interact with the underlying physical media driver. This command can be applied to a single modem or a group of modems.

Once the **debug dsip trace** command has been enabled, you can read the information captured in the trace buffer using the **show dsip tracing** command.

**Examples**

The following example shows the available **debug dsip** command options:

```
AS5800> enable
Password: letmein
AS5800# debug dsip ?
  all          All DSIP debugging messages
  api          DSIP API debugging
  boot        DSIP booting
  console     DSIP console
  trace       DSIP tracing
  transport   DSIP transport
```

The following example indicates the **debug dsip trace** command logs MAC headers of the various classes of DSIP packets. View the logged information using the **show dsip tracing** command:

```
AS5800# debug dsip trace
NIP tracing debugging is on
AS5800# show dsip tracing
NIP Control Packet Trace
-----
Dest:00e0.b093.2238 Src:0007.4c72.0058 Type:200B SrcShelf:1 SrcSlot:11
MsgType:0 MsgLen:82 Timestamp: 00:49:14
-----
Dest:00e0.b093.2238 Src:0007.4c72.0028 Type:200B SrcShelf:1 SrcSlot:5
MsgType:0 MsgLen:82 Timestamp: 00:49:14
-----
```

**Related Commands**

Command	Description
<b>debug modem dsip</b>	Displays information about the dial shelf, including clocking information.
<b>show dsip tracing</b>	Displays DSIP media header information logged using the <b>debug dsip trace</b> command.

# debug dspapi

To enable debugging for Digital Signal Processor (DSP) Application Programming Interface (API) message events, use the **debug dspapi** command in privileged EXEC mode. To reset the default value for this feature, use the **no** form of this command.

**debug dspapi** { **all** | **command** | **detail** | **error** | **notification** | **response** }

[**no**] **debug dspapi** { **all** | **command** | **detail** | **error** | **notification** | **response** }

Syntax Description	all	Enables all dspapi debug options (command, detail, error, notification and response).
	<b>command</b>	Displays commands sent to the DSPs.
	<b>detail</b>	Displays additional detail for the DSP API debugs enabled.
	<b>error</b>	Displays any dspapi errors.
	<b>notification</b>	Displays notification messages sent from the DSP (for example, tone detection notification).
	<b>response</b>	Displays responses sent by the DSP (for example, responses to statistic requests).

**Defaults** This command is not enabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)XM	This command was introduced on the Cisco AS5300 and Cisco AS5800.
	12.1(5)XM1	This command was implemented on the Cisco AS 5350 and Cisco AS5400.
	12.2(2)T	This command was implemented on the Cisco 1700, Cisco 2600 series, Cisco 3600 series, and the Cisco 3810.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.

**Usage Guidelines** DSP API message events used to communicate with DSPs are intended for use with Connexant (Nextport) and Texas Instrument (54x) DSPs. This command severely impacts performance and should be used only for single-call debug capture.

**Examples** The following example shows how to enable debugging for all DSP API message events:

```
Router debug dspapi all
```

■ debug dspapi

---

**Related Commands**

Command	Description
<b>debug hpi</b>	Enables debugging for Host Port Interface (HPI) message events.

---

# debug dspfarm

To display digital signal processor (DSP) farm service debugging information, use the **debug dspfarm** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

**debug dspfarm** { **all** | **errors** | **events** | **packets** }

**no debug dspfarm**

Syntax Description	all	All DSP-farm debug-trace information.
	<b>errors</b>	DSP-farm errors.
	<b>events</b>	DSP-farm events.
	<b>packets</b>	DSP-farm packets.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)YH	This command was introduced on the Cisco VG200.
	12.2(13)T	This command was implemented on the Cisco 2600 series, Cisco 3620, Cisco 3640, Cisco 3660, and Cisco 3700 series.

**Usage Guidelines** The router on which this command is used must be equipped with one or more digital T1/E1 packet voice trunk network modules (NM-HDVs) or high-density voice (HDV) transcoding/conferencing DSP farms (NM-HDV-FARMS) to provide DSP resources.

Debugging is turned on for all DSP-farm-service sessions. You can debug multiple sessions simultaneously, with different levels of debugging for each.

**Examples** The following is sample output from the **debug dspfarm events** command:

```
Router# debug dspfarm events
```

```
DSP Farm service events debugging is on
```

```
*Mar 1 00:45:51: Sent 180 bytes to DSP 4 channel 2
*Mar 1 00:45:53: Sent 180 bytes to DSP 4 channel 3
*Mar 1 00:45:55: Sent 180 bytes to DSP 4 channel 1
*Mar 1 00:45:56: Sent 180 bytes to DSP 4 channel 2
*Mar 1 00:45:58: Sent 180 bytes to DSP 4 channel 3
*Mar 1 00:46:00: Sent 180 bytes to DSP 4 channel 1
*Mar 1 00:46:01: xapi_dspfarm_modify_connection: sess_id 26, conn_id 2705, conn_mode 3,
ripaddr 10.10.1.7, rport 20170
*Mar 1 00:46:01: dspfarm_process_appl_event_queue: XAPP eve 6311C4B0 rcvd
```

```

*Mar 1 00:46:01: dspfarm_find_stream: stream 63121F1C, found in sess 631143CC, cid 2705
*Mar 1 00:46:01: dspfarm_modify_connection: old_mode 4, new_mode 3
*Mar 1 00:46:01: dspfarm_close_local_rtp: stream 63121F1C, local_rtp_port 22656
*Mar 1 00:46:01: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C4C8,
eve_id 5, context 6311426C, result 0
*Mar 1 00:46:01: xapi_dspfarm_delete_connection: sess_id 26, conn_id 2705
*Mar 1 00:46:01: dspfarm_process_appl_event_queue: XAPP eve 6311C4E0 rcvd
*Mar 1 00:46:01: dspfarm_find_stream: stream 63121F1C, found in sess 631143CC, cid 2705
*Mar 1 00:46:01: dspfarm_close_local_rtp: stream 63121F1C, local_rtp_port 0
*Mar 1 00:46:01: dspfarm_release_dsp_resource: sess 631143CC, stream 63121F1C, num_stream
3, sess_type 2, sess_dsp_id 2040000, stream_dsp_id 2040002
*Mar 1 00:46:01: dspfarm_drop_conference:slot 2 dsp 4 ch 2
*Mar 1 00:46:01: dspfarm_send_drop_conf: Sent drop_conference to DSP 4 ch 2
*Mar 1 00:46:01: dspfarm_xapp_enq: Sent msg 8 to DSPFARM
*Mar 1 00:46:01: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C4F8,
eve_id 9, context 6311426C, result 0
*Mar 1 00:46:01: dspfarm_process_dsp_event_queue: DSP eve 6312078C rcvd
*Mar 1 00:46:01: dspfarm_delete_stream: sess_id 26, conn_id 2705, stream 63121F1C, in
sess 631143CC is freed
*Mar 1 00:46:01: Sent 180 bytes to DSP 4 channel 3
*Mar 1 00:46:04: Sent 180 bytes to DSP 4 channel 3
*Mar 1 00:46:05: xapi_dspfarm_modify_connection: sess_id 26, conn_id 2689, conn_mode 3,
ripaddr 10.10.1.5, rport 19514
*Mar 1 00:46:05: dspfarm_process_appl_event_queue: XAPP eve 6311C510 rcvd
*Mar 1 00:46:05: dspfarm_find_stream: stream 63121E34, found in sess 631143CC, cid 2689
*Mar 1 00:46:05: dspfarm_modify_connection: old_mode 4, new_mode 3
*Mar 1 00:46:05: dspfarm_close_local_rtp: stream 63121E34, local_rtp_port 25834
*Mar 1 00:46:05: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C528,
eve_id 5, context 63114244, result 0
*Mar 1 00:46:05: xapi_dspfarm_delete_connection: sess_id 26, conn_id 2689
*Mar 1 00:46:05: dspfarm_process_appl_event_queue: XAPP eve 6311C540 rcvd
*Mar 1 00:46:05: dspfarm_find_stream: stream 63121E34, found in sess 631143CC, cid 2689
*Mar 1 00:46:05: dspfarm_close_local_rtp: stream 63121E34, local_rtp_port 0
*Mar 1 00:46:05: dspfarm_release_dsp_resource: sess 631143CC, stream 63121E34, num_stream
2, sess_type 2, sess_dsp_id 2040000, stream_dsp_id 2040001
*Mar 1 00:46:05: dspfarm_drop_conference:slot 2 dsp 4 ch 1
*Mar 1 00:46:05: dspfarm_send_drop_conf: Sent drop_conference to DSP 4 ch 1
*Mar 1 00:46:05: dspfarm_xapp_enq: Sent msg 8 to DSPFARM
*Mar 1 00:46:05: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C558,
eve_id 9, context 63114244, result 0
*Mar 1 00:46:05: dspfarm_process_dsp_event_queue: DSP eve 6311586C rcvd
*Mar 1 00:46:05: dspfarm_delete_stream: sess_id 26, conn_id 2689, stream 63121E34, in
sess 631143CC is freed
*Mar 1 00:46:05: xapi_dspfarm_modify_connection: sess_id 26, conn_id 2721, conn_mode 3,
ripaddr 10.10.1.6, rport 21506
*Mar 1 00:46:05: dspfarm_process_appl_event_queue: XAPP eve 6311C570 rcvd
*Mar 1 00:46:05: dspfarm_find_stream: stream 63122004, found in sess 631143CC, cid 2721
*Mar 1 00:46:05: dspfarm_modify_connection: old_mode 4, new_mode 3
*Mar 1 00:46:05: dspfarm_close_local_rtp: stream 63122004, local_rtp_port 19912
*Mar 1 00:46:05: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C588,
eve_id 5, context 63114294, result 0
*Mar 1 00:46:05: xapi_dspfarm_delete_connection: sess_id 26, conn_id 2721
*Mar 1 00:46:05: dspfarm_process_appl_event_queue: XAPP eve 6311C5A0 rcvd
*Mar 1 00:46:05: dspfarm_find_stream: stream 63122004, found in sess 631143CC, cid 2721
*Mar 1 00:46:05: dspfarm_close_local_rtp: stream 63122004, local_rtp_port 0
*Mar 1 00:46:05: dspfarm_release_dsp_resource: sess 631143CC, stream 63122004, num_stream
1, sess_type 2, sess_dsp_id 2040000, stream_dsp_id 2040003
*Mar 1 00:46:05: dspfarm_drop_conference:slot 2 dsp 4 ch 3
*Mar 1 00:46:05: dspfarm_drop_conference: Last conferee - closing the conf session
*Mar 1 00:46:05: dspfarm_send_close_conf: Sent close_conference to DSP 4
*Mar 1 00:46:05: dspfarm_drop_conference: Removed the conf in dsp 4
*Mar 1 00:46:05: dspfarm_xapp_enq: Sent msg 8 to DSPFARM
*Mar 1 00:46:05: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C5B8,
eve_id 9, context 63114294, result 0

```

```
*Mar 1 00:46:05: dspfarm_process_dsp_event_queue: DSP eve 6311586C rcvd
*Mar 1 00:46:05: dspfarm_delete_stream: sess_id 26, conn_id 2721, stream 63122004, in
sess 631143CC is freed
```

---

**Related Commands**

Command	Description
<b>debug frame-relay vc-bundle</b>	Sets debugging for SCCP and its applications at one of four levels.
<b>dspfarm (DSP farm)</b>	Enables DSP-farm service.
<b>sccp</b>	Enables SCCP and its associated transcoding and conferencing applications.
<b>show dspfarm</b>	Displays summary information about DSP resources.

# debug dspu activation

To display information on downstream physical unit (DSPU) activation, use the **debug dspu activation** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug dspu activation** [*name*]

**no debug dspu activation** [*name*]

<b>Syntax Description</b>	<i>name</i> (Optional) The host or physical unit (PU) name designation.
---------------------------	---

<b>Command Modes</b>	Privileged EXEC
----------------------	-----------------

<b>Usage Guidelines</b>	The <b>debug dspu activation</b> command displays all DSPU activation traffic. To restrict the output to a specific host or PU, include the host or PU <i>name</i> argument. You cannot turn off debugging output for an individual PU if that PU has not been named in the <b>debug dspu activation</b> command.
-------------------------	---

<b>Examples</b>	The following is sample output from the <b>debug dspu activation</b> command. Not all intermediate numbers are shown for the “activated” and “deactivated” logical unit (LU) address ranges.
-----------------	--

```
Router# debug dspu activation

DSPU: LS HOST3745 connected
DSPU: PU HOST3745 activated
DSPU: LU HOST3745-2 activated
DSPU: LU HOST3745-3 activated
.
.
.
DSPU: LU HOST3745-253 activated
DSPU: LU HOST3745-254 activated

DSPU: LU HOST3745-2 deactivated
DSPU: LU HOST3745-3 deactivated
.
.
.
DSPU: LU HOST3745-253 deactivated
DSPU: LU HOST3745-254 deactivated
DSPU: LS HOST3745 disconnected
DSPU: PU HOST3745 deactivated
```

Table 56 describes the significant fields shown in the display.

**Table 56** *debug dspu activation Command Field Descriptions*

Field	Description
DSPU	Downstream PU debug message.
LS	Link station (LS) event triggered the message.
PU	PU event triggered the message.
LU	LU event triggered the message.
HOST3745	Host name or PU name.
HOST3745-253	Host name or PU name and the LU address, separated by a dash.
connected activated disconnected deactivated	Event that occurred to trigger the message.

#### Related Commands

Command	Description
<a href="#">debug dspu packet</a>	Displays information on a DSPU packet.
<a href="#">debug dspu state</a>	Displays information on DSPU FSM state changes.
<a href="#">debug dspu trace</a>	Displays information on DSPU trace activity.

# debug dspu packet

To display information on a downstream physical unit (DSPU) packet, use the **debug dspu packet** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug dspu packet** [*name*]

**no debug dspu packet** [*name*]

## Syntax Description

*name* (Optional) The host or PU name designation.

## Command Modes

Privileged EXEC

## Usage Guidelines

The **debug dspu packet** command displays all DSPU packet data flowing through the router. To restrict the output to a specific host or PU, include the host or PU *name* argument. You cannot turn off debugging output for an individual PU if that PU has not been named in the **debug dspu packet** command.

## Examples

The following is sample output from the **debug dspu packet** command:

```
Router# debug dspu packet

DSPU: Rx: PU HOST3745 data length 12 data:
      2D0003002BE16B80 000D0201
DSPU: Tx: PU HOST3745 data length 25 data:
      2D0000032BE1EB80 000D020100850000 000C060000010000 00
DSPU: Rx: PU HOST3745 data length 12 data:
      2D0004002BE26B80 000D0201
DSPU: Tx: PU HOST3745 data length 25 data:
      2D0000042BE2EB80 000D020100850000 000C060000010000 00
```

[Table 57](#) describes the significant fields shown in the display.

**Table 57** *debug dspu packet* Command Field Descriptions

Field	Description
DSPU: Rx:	Received frame (packet) from the remote PU to the router PU.
DSPU: Tx:	Transmitted frame (packet) from the router PU to the remote PU.
PU HOST3745	Host name or PU associated with the transmit or receive.
data length 12 data:	Number of bytes of data, followed by up to 128 bytes of displayed data.

## Related Commands

Command	Description
<a href="#">debug drip event</a>	Displays debug messages for DRiP packets.
<a href="#">debug dspu state</a>	Displays information on DSPU FSM state changes.
<a href="#">debug dspu trace</a>	Displays information on DSPU trace activity.

# debug dspu state

To display information on downstream physical unit (DSPU) finite state machine (FSM) state changes, use the **debug dspu state** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug dspu state** [*name*]

**no debug dspu state** [*name*]

## Syntax Description

*name* (Optional) The host or PU name designation.

## Command Modes

Privileged EXEC

## Usage Guidelines

Use the **debug dspu state** command to display only the FSM state changes. To see all FSM activity, use the **debug dspu trace** command. You cannot turn off debugging output for an individual PU if that PU has not been named in the **debug dspu state** command.

## Examples

The following is sample output from the **debug dspu state** command. Not all intermediate numbers are shown for the “activated” and “deactivated” logical unit (LU) address ranges.

```
Router# debug dspu state

DSPU: LS HOST3745: input=StartLs, Reset -> PendConOut
DSPU: LS HOST3745: input=ReqOpn.Cnf, PendConOut -> Xid
DSPU: LS HOST3745: input=Connect.Ind, Xid -> ConnIn
DSPU: LS HOST3745: input=Connected.Ind, ConnIn -> Connected
DSPU: PU HOST3745: input=Actpu, Reset -> Active
DSPU: LU HOST3745-2: input=uActlu, Reset -> upLuActive
DSPU: LU HOST3745-3: input=uActlu, Reset -> upLuActive
.
.
.
DSPU: LU HOST3745-253: input=uActlu, Reset -> upLuActive
DSPU: LU HOST3745-254: input=uActlu, Reset -> upLuActive

DSPU: LS HOST3745: input=PuStopped, Connected -> PendDisc
DSPU: LS HOST3745: input=Disc.Cnf, PendDisc -> PendClose
DSPU: LS HOST3745: input=Close.Cnf, PendClose -> Reset
DSPU: PU HOST3745: input=T2ResetPu, Active -> Reset
DSPU: LU HOST3745-2: input=uStopLu, upLuActive -> Reset
DSPU: LU HOST3745-3: input=uStopLu, upLuActive -> Reset
.
.
.
DSPU: LU HOST3745-253: input=uStopLu, upLuActive -> Reset
DSPU: LU HOST3745-254: input=uStopLu, upLuActive -> Reset
```

Table 58 describes the significant fields shown in the display.

**Table 58** *debug dspu state* Command Field Descriptions

Field	Description
DSPU	Downstream PU debug message.
LS	Link station (LS) event triggered the message.
PU	PU event triggered the message.
LU	LU event triggered the message.
HOST3745-253	Host name or PU name and LU address.
input=input,	Input received by the FSM.
previous-state, -> current-state	Previous state and current new state as seen by the FSM.

#### Related Commands

Command	Description
<b>debug drip event</b>	Displays debug messages for DRiP packets.
<b>debug drip packet</b>	Displays information on DSPU packet.
<b>debug dspu trace</b>	Displays information on DSPU trace activity.

# debug dspu trace

To display information on downstream physical unit (DSPU) trace activity, which includes all finite state machine (FSM) activity, use the **debug dspu trace** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug dspu trace** [*name*]

**no debug dspu trace** [*name*]

## Syntax Description

*name* (Optional) The host or PU name designation.

## Command Modes

Privileged EXEC

## Usage Guidelines

Use the **debug dspu trace** command to display all FSM state changes. To see FSM state changes only, use the **debug dspu state** command. You cannot turn off debugging output for an individual PU if that PU has not been named in the **debug dspu trace** command.

## Examples

The following is sample output from the **debug dspu trace** command:

```
Router# debug dspu trace

DSPU: LS HOST3745 input = 0 ->(1,a1)
DSPU: LS HOST3745 input = 5 ->(5,a6)
DSPU: LS HOST3745 input = 7 ->(5,a9)
DSPU: LS HOST3745 input = 9 ->(5,a28)
DSPU: LU HOST3745-2 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
DSPU: LS HOST3745 input = 18 ->(8,a17)
DSPU: LU HOST3745-3 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
DSPU: LS HOST3745 input = 18 ->(8,a17)
DSPU: LU HOST3745-252 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
DSPU: LS HOST3745 input = 18 ->(8,a17)
DSPU: LU HOST3745-253 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
DSPU: LS HOST3745 input = 18 ->(8,a17)
DSPU: LU HOST3745-254 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
```

Table 59 describes significant fields in the output.

**Table 59** *debug dspu trace Command Field Descriptions*

Field	Description
7:23:57	Time stamp.
DSPU	Downstream PU debug message.
LS	Link station (LS) event triggered the message.
PU	A PU event triggered the message.
LU	LU event triggered the message.
HOST3745-253	Host name or PU name and LU address.
in:input s:state ->(new-state, action)	String describing the following: <ul style="list-style-type: none"> <li>input—LU FSM input</li> <li>state—Current FSM state</li> <li>new-state—New FSM state</li> <li>action—FSM action</li> </ul>
input=input -> (new-state, action)	String describing the following: <ul style="list-style-type: none"> <li>input—PU or LS FSM input</li> <li>new-state—New PU or LS FSM state</li> <li>action—PU or LS FSM action</li> </ul>

#### Related Commands

Command	Description
<a href="#">debug drip event</a>	Displays debug messages for DRiP packets.
<a href="#">debug drip packet</a>	Displays information on DSPU packet.
<a href="#">debug dspu state</a>	Displays information on DSPU FSM state changes.

# debug dss ipx event

To display debug messages for route change events that affect IPX Multilayer Switching (MLS), use the **debug dss ipx event** command in privileged EXEC mode. To disable debugging output, use the **no** form of the command.

**debug dss ipx event**

**no debug dss ipx event**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debugging is not enabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(5)T	This command was introduced.

**Examples** The following displays sample output from the **debug dss ipx event** command:

```
Router# debug dss ipx event

DSS IPX events debugging is on
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# interface vlan 22
Router(config-if)# ipx access-group 800 out
05:51:36:DSS-feature:dss_ipxcache_version():idb:NULL, reason:42,
prefix:0, mask:FFFFFFFF
05:51:36:DSS-feature:dss_ipx_access_group():idb:Vlan22
05:51:36:DSS-feature:dss_ipx_access_list()
05:51:36:DSS-base 05:51:33.834 dss_ipx_invalidate_interface V122
05:51:36:DSS-base 05:51:33.834 dss_set_ipx_flowmask_reg 2
05:51:36:%IPX mls flowmask transition from 1 to 2 due to new status of
simple IPX access list on interfaces
```

Related Commands	Command	Description
	<b>debug mls rp</b>	Displays various MLS debugging elements.

# debug eigrp fsm

To display debugging information about Enhanced Interior Gateway Routing Protocol (EIGRP) feasible successor metrics (FSM), use the **debug eigrp fsm** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug eigrp fsm**

**no debug eigrp fsm**

---

**Syntax Description** This command has no arguments or keywords.

---

**Defaults** No default behavior or values.

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command helps you observe EIGRP feasible successor activity and to determine whether route updates are being installed and deleted by the routing process.

---

**Examples** The following is sample output from the **debug eigrp fsm** command:

```
Router# debug eigrp fsm

DUAL: dual_rcvupdate(): 172.25.166.0 255.255.255.0 via 0.0.0.0 metric 750080/0
DUAL: Find FS for dest 172.25.166.0 255.255.255.0. FD is 4294967295, RD is 42949
67295 found
DUAL: RT installed 172.25.166.0 255.255.255.0 via 0.0.0.0
DUAL: dual_rcvupdate(): 192.168.4.0 255.255.255.0 via 0.0.0.0 metric 4294967295/
4294967295
DUAL: Find FS for dest 192.168.4.0 255.255.255.0. FD is 2249216, RD is 2249216
DUAL: 0.0.0.0 metric 4294967295/4294967295not found Dmin is 4294967295
DUAL: Dest 192.168.4.0 255.255.255.0 not entering active state.
DUAL: Removing dest 192.168.4.0 255.255.255.0, nexthop 0.0.0.0
DUAL: No routes. Flushing dest 192.168.4.0 255.255.255.0
```

In the first line, DUAL stands for diffusing update algorithm. It is the basic mechanism within EIGRP that makes the routing decisions. The next three fields are the Internet address and mask of the destination network and the address through which the update was received. The metric field shows the metric stored in the routing table and the metric advertised by the neighbor sending the information. If shown, the term “Metric... inaccessible” usually means that the neighbor router no longer has a route to the destination, or the destination is in a hold-down state.

In the following output, EIGRP is attempting to find a feasible successor for the destination. Feasible successors are part of the DUAL loop avoidance methods. The FD field contains more loop avoidance state information. The RD field is the reported distance, which is the metric used in update, query, or reply packets.

The indented line with the “not found” message means a feasible successor (FS) was not found for 192.168.4.0 and EIGRP must start a diffusing computation. This means it begins to actively probe (sends query packets about destination 192.168.4.0) the network looking for alternate paths to 192.164.4.0.

```
DUAL: Find FS for dest 192.168.4.0 255.255.255.0. FD is 2249216, RD is 2249216
DUAL:   0.0.0.0 metric 4294967295/4294967295not found Dmin is 4294967295
```

The following output indicates the route DUAL successfully installed into the routing table:

```
DUAL: RT installed 172.25.166.0 255.255.255.0 via 0.0.0.0
```

The following output shows that no routes to the destination were discovered and that the route information is being removed from the topology table:

```
DUAL: Dest 192.168.4.0 255.255.255.0 not entering active state.
DUAL: Removing dest 192.168.4.0 255.255.255.0, nexthop 0.0.0.0
DUAL: No routes. Flushing dest 192.168.4.0 255.255.255.0
```

# debug eigrp neighbor

To display neighbors discovered by the Enhanced Interior Gateway Routing Protocol (EIGRP), use the **debug eigrp neighbor** command in privileged EXEC mode. To disable **debug eigrp neighbor**, use the **no** form of this command.

**debug eigrp neighbor [siatimer] [static]**

**no debug eigrp neighbor [siatimer] [static]**

Syntax Description	Parameter	Description
	<b>siatimer</b>	(Optional) Stuck-in-active (SIA) timer messages.
	<b>static</b>	(Optional) Static routes.

**Defaults** Debugging for EIGRP neighbors is not enabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(7)T	This command was introduced.

**Examples** The following is sample output from the **debug eigrp neighbor** command.

```
Router# debug eigrp neighbor static

EIGRP Static Neighbors debugging is on

Router#configure terminal

Router(config)#router eigrp 100

Router(config-router)#neighbor 10.1.1.1 e3/1

Router(config-router)#
22:40:07:EIGRP:Multicast Hello is disabled on Ethernet3/1!
22:40:07:EIGRP:Add new static nbr 10.1.1.1 to AS 100 Ethernet3/1

Router(config-router)#no neighbor 10.1.1.1 e3/1

Router(config-router)#
22:41:23:EIGRP:Static nbr 10.1.1.1 not in AS 100 Ethernet3/1 dynamic list
22:41:23:EIGRP>Delete static nbr 10.1.1.1 from AS 100 Ethernet3/1
22:41:23:EIGRP:Multicast Hello is enabled on Ethernet3/1!
```

---

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>show ip eigrp neighbors</b>	Displays EIGRP neighbors.
<b>neighbor</b>	Defines a neighboring router with which to exchange routing information.

# debug eigrp nsf

To display nonstop forwarding (NSF) events in the console of the router, use the **debug eigrp nsf** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

**debug eigrp nsf**

**no debug eigrp nsf**

---

**Syntax Description** This command has no arguments or keywords.

---

**Defaults** No default behavior or values.

---

**Command Modes** Privileged EXEC

---

Command History	Release	Modification
	12.2(15)T	This command was introduced.

---



---

**Usage Guidelines** The output from the **debug eigrp nsf** command displays NSF-specific events. This command can be issued on an NSF-capable or NSF-aware router.

---

**Examples** The following example enables Enhanced Interior Gateway Routing Protocol (EIGRP) NSF debugging:

```
Router# debug eigrp nsf
```

---

Related Commands	Command	Description
	<b>timers nsf route-hold</b>	Sets the route-hold timer for NSF-aware routers that run EIGRP.

---

# debug eigrp packet

To display general debugging information, use the **debug eigrp packet** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug eigrp packet**

**no debug eigrp packet**

---

**Syntax Description** This command has no arguments or keywords.

---

**Defaults** No default behavior or values.

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** If a communication session is closing when it should not be, an end-to-end connection problem can be the cause. The **debug eigrp packet** command is useful for analyzing the messages traveling between the local and remote hosts.

---

**Examples** The following is sample output from the **debug eigrp packet** command:

```
Router# debug eigrp packet

EIGRP: Sending HELLO on Ethernet0/1
      AS 109, Flags 0x0, Seq 0, Ack 0
EIGRP: Sending HELLO on Ethernet0/1
      AS 109, Flags 0x0, Seq 0, Ack 0
EIGRP: Sending HELLO on Ethernet0/1
      AS 109, Flags 0x0, Seq 0, Ack 0
EIGRP: Received UPDATE on Ethernet0/1 from 192.195.78.24,
      AS 109, Flags 0x1, Seq 1, Ack 0
EIGRP: Sending HELLO/ACK on Ethernet0/1 to 192.195.78.24,
      AS 109, Flags 0x0, Seq 0, Ack 1
EIGRP: Sending HELLO/ACK on Ethernet0/1 to 192.195.78.24,
      AS 109, Flags 0x0, Seq 0, Ack 1
EIGRP: Received UPDATE on Ethernet0/1 from 192.195.78.24,
      AS 109, Flags 0x0, Seq 2, Ack 0
```

The output shows transmission and receipt of Enhanced Interior Gateway Routing Protocol (EIGRP) packets. These packet types may be hello, update, request, query, or reply packets. The sequence and acknowledgment numbers used by the EIGRP reliable transport algorithm are shown in the output. Where applicable, the network-layer address of the neighboring router is also included.

Table 60 describes the significant fields shown in the display.

**Table 60** *debug eigrp packet Command Field Descriptions*

Field	Description
EIGRP:	EIGRP packet information.
AS n	Autonomous system number.
Flags <i>nxn</i>	<p>A flag of 1 means the sending router is indicating to the receiving router that this is the first packet it has sent to the receiver.</p> <p>A flag of 2 is a multicast that should be conditionally received by routers that have the conditionally receive (CR) bit set. This bit gets set when the sender of the multicast has previously sent a sequence packet explicitly telling it to set the CR bit.</p>
HELLO	<p>Hello packets are the neighbor discovery packets. They are used to determine whether neighbors are still alive. As long as neighbors receive the hello packets the router is sending, the neighbors validate the router and any routing information sent. If neighbors lose the hello packets, the receiving neighbors invalidate any routing information previously sent. Neighbors also send hello packets.</p>

# debug eigrp transmit

To display transmittal messages sent by the Enhanced Interior Gateway Routing Protocol (EIGRP), use the **debug eigrp transmit** command in privileged EXEC mode. To disable **debug eigrp transmit**, use the **no** form of this command.

**debug eigrp transmit** [**ack**] [**build**] [**detail**] [**link**] [**packetize**] [**peerdown**] [**startup**] [**strange**]

**no debug eigrp transmit** [**ack**] [**build**] [**detail**] [**link**] [**packetize**] [**peerdown**] [**sia**] [**startup**] [**strange**]

Syntax Description	
<b>ack</b>	(Optional) Information for acknowledgment (ACK) messages sent by the system.
<b>build</b>	(Optional) Build information messages (messages that indicate that a topology table was either successfully built or could not be built).
<b>detail</b>	(Optional) Additional detail for debug output.
<b>link</b>	(Optional) Information regarding topology table linked-list management.
<b>packetize</b>	(Optional) Information regarding topology table linked-list management.
<b>peerdown</b>	(Optional) Information regarding the impact on packet generation when a peer is down.
<b>sia</b>	(Optional) Stuck-in-active (SIA) messages.
<b>startup</b>	(Optional) Information regarding peer startup and initialization packets that have been transmitted.
<b>strange</b>	(Optional) Unusual events relating to packet processing.

**Defaults** Debugging for Enhanced IGRP transmittal messages is not enabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1	This command was introduced.

---

**Examples**

The following is sample output from the **debug eigrp transmit** command.

```
Router# debug eigrp transmit

EIGRP Transmission Events debugging is on
  (ACK, PACKETIZE, STARTUP, PEERDOWN, LINK, BUILD, STRANGE, SIA, DETAIL)

Router#configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.
Router#(config)#router eigrp 100
Router#(config-router)#network 10.4.9.0 0.0.0.255
Router#(config-router)#
5d22h: DNDB UPDATE 10.0.0.0/8, serno 0 to 1, refcount 0
Router#(config-router)#
```

# debug ephone alarm

To set SkinnyStation alarm messages debugging for the Cisco IP phone, use the **debug ephone alarm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone alarm** [**mac-address** *mac-address*]

**no debug ephone alarm** [**mac-address** *mac-address*]

Syntax Description	mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
	<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.2(2)XT	This command was introduced on the following platforms: Cisco 1750, Cisco 1751, Cisco 2600 series and Cisco 3600 series multiservice routers; and Cisco IAD2420 series Integrated Access Devices (IADs).
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone alarm** command shows all the SkinnyStation alarm messages sent by the Cisco IP phone. Under normal circumstances, this message is sent by the Cisco IP phone just before it registers, and the message has the severity level for the alarm set to “Informational” and contains the reason for the phone reboot or re-register. This type of message is entirely benign and does not indicate an error condition.

If the **mac-address** keyword is not used, the **debug ephone alarm** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a SkinnyStation alarm message that is sent before the Cisco IP phone registers:

```
Router# debug ephone alarm

phone keypad reset
CM-closed-TCP
CM-bad-state
```

**Related Commands**

Command	Description
<a href="#">debug ephone detail</a>	Sets detail debugging for the Cisco IP phone.
<a href="#">debug ephone error</a>	Sets error debugging for the Cisco IP phone.
<a href="#">debug ephone keepalive</a>	Sets keepalive debugging for the Cisco IP phone.
<a href="#">debug ephone mwi</a>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<a href="#">debug ephone pak</a>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<a href="#">debug ephone raw</a>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<a href="#">debug ephone register</a>	Sets registration debugging for the Cisco IP phone.
<a href="#">debug ephone state</a>	Sets state debugging for the Cisco IP phone.
<a href="#">debug ephone statistics</a>	Sets statistics debugging for the Cisco IP phone.
<a href="#">show debugging</a>	Displays information about the types of debugging that are enabled for your router.

# debug ephone detail

To set detail debugging for the Cisco IP phone, use the **debug ephone detail** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone detail** [**mac-address** *mac-address*]

**no debug ephone detail** [**mac-address** *mac-address*]

Syntax Description	mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
	<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
	12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone detail** command includes the error and state levels.

If the **mac-address** keyword is not used, the **debug ephone detail** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a sample output of detail debugging of the Cisco IP phone with MAC address 0030.94c3.8724. The sample is an excerpt of some of the activities that takes place during call setup, connected state, active call, and the call being disconnected:

```
Router# debug ephone detail mac-address 0030.94c3.8724
Ephone detail debugging is enabled

1d04h: ephone-1[1]:OFFHOOK
.
.
1d04h: Skinny Call State change for DN 1 SIEZE
.
.
1d04h: ephone-1[1]:SetCallState line 1 DN 1 TsOffHook
.
.
1d04h: ephone-1[1]:SetLineLamp 1 to ON
.
.
1d04h: ephone-1[1]:KeypadButtonMessage 5
.
.
1d04h: ephone-1[1]:KeypadButtonMessage 0
.
.
1d04h: ephone-1[1]:KeypadButtonMessage 0
.
.
1d04h: ephone-1[1]:KeypadButtonMessage 2
.
.
1d04h: ephone-1[1]:Store ReDial digit: 5002
.
SkinnyTryCall to 5002 instance 1
.
.
1d04h: ephone-1[1]:Store ReDial digit: 5002
1d04h: ephone-1[1]:
SkinnyTryCall to 5002 instance 1
.
.
1d04h: Skinny Call State change for DN 1 ALERTING
.
.
1d04h: ephone-1[1]:SetCallState line 1 DN 1 TsRingOut
.
.
1d04h: ephone-1[1]:SetLineLamp 1 to ON
1d04h: SetCallInfo calling dn 1 dn 1
calling [5001] called [5002]
.
.
1d04h: ephone-1[1]: Jane calling
1d04h: ephone-1[1]: Jill
.
.
1d04h: SkinnyUpdateDnState by EFXS_RING_GENERATE
for DN 2 to state RINGING
.
.
1d04h: SkinnyGetCallState for DN 2 CONNECTED
.
.
```

```

1d04h: ephone-1[1]:SetLineLamp 3 to ON
1d04h: ephone-1[1]:UpdateCallState DN 1 state 4 calleddn 2
.
.
1d04h: Skinny Call State change for DN 1 CONNECTED
.
.
1d04h: ephone-1[1]:OpenReceive DN 1 codec 4:G711Ulaw64k duration 10 ms bytes 80
.
.
1d04h: ephone-1[1]:OpenReceiveChannelAck 1.2.172.21 port=20180
1d04h: ephone-1[1]:Outgoing calling DN 1 Far-ephone-2 called DN 2
1d04h: SkinnyGetCallState for DN 1 CONNECTED
.
.
1d04h: ephone-1[1]:SetCallState line 3 DN 2 TsOnHook
.
.
1d04h: ephone-1[1]:SetLineLamp 3 to OFF
.
.
1d04h: ephone-1[1]:SetCallState line 1 DN 1 TsOnHook
.
.
1d04h: ephone-1[1]:Clean Up Speakerphone state
1d04h: ephone-1[1]:SpeakerPhoneOnHook
1d04h: ephone-1[1]:Clean up activeline 1
1d04h: ephone-1[1]:StopTone sent to ephone
1d04h: ephone-1[1]:Clean Up phone offhook state
1d04h: SkinnyGetCallState for DN 1 IDLE
1d04h: called DN -1, calling DN -1 phone -1
1d04h: ephone-1[1]:SetLineLamp 1 to OFF
1d04h: UnBinding ephone-1 from DN 1
1d04h: UnBinding called DN 2 from DN 1
1d04h: ephone-1[1]:ONHOOK
1d04h: ephone-1[1]:SpeakerPhoneOnHook
1d04h: ephone-1[1]:ONHOOK NO activeline
.
.
.

```

#### Related Commands

Command	Description
<a href="#">debug ephone alarm</a>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<a href="#">debug ephone error</a>	Sets error debugging for the Cisco IP phone.
<a href="#">debug ephone keepalive</a>	Sets keepalive debugging for the Cisco IP phone.
<a href="#">debug ephone mwi</a>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<a href="#">debug ephone pak</a>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<a href="#">debug ephone raw</a>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<a href="#">debug ephone register</a>	Sets registration debugging for the Cisco IP phone.
<a href="#">debug ephone state</a>	Sets state debugging for the Cisco IP phone.

Command	Description
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone error

To set error debugging for the Cisco IP phone, use the **debug ephone error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone error** [**mac-address** *mac-address*]

**no debug ephone error** [**mac-address** *mac-address*]

Syntax Description	Parameter	Description
	<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
	<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
	12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone error** command cancels debugging at the detail and state level.

If the **mac-address** keyword is not used, the **debug ephone error** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a sample output of error debugging for the Cisco IP phone with MAC address 0030.94c3.8724:

```
Router# debug ephone error mac-address 0030.94c3.8724
```

```
EPHONE error debugging is enabled
```

```
socket [2] send ERROR 11
Skinny Socket [2] retry failure
```

**Related Commands**

Command	Description
<a href="#">debug ephone alarm</a>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<a href="#">debug ephone detail</a>	Sets detail debugging for the Cisco IP phone.
<a href="#">debug ephone keepalive</a>	Sets keepalive debugging for the Cisco IP phone.
<a href="#">debug ephone mwi</a>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<a href="#">debug ephone pak</a>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<a href="#">debug ephone raw</a>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<a href="#">debug ephone register</a>	Sets registration debugging for the Cisco IP phone.
<a href="#">debug ephone state</a>	Sets state debugging for the Cisco IP phone.
<a href="#">debug ephone statistics</a>	Sets statistics debugging for the Cisco IP phone.
<a href="#">show debugging</a>	Displays information about the types of debugging that are enabled for your router.

# debug ephone keepalive

To set keepalive debugging for the Cisco IP phone, use the **debug ephone keepalive** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone keepalive** [**mac-address** *mac-address*]

**no debug ephone keepalive** [**mac-address** *mac-address*]

Syntax Description	Parameter	Description
	<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
	<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
	12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone keepalive** command sets keepalive debugging.

If the **mac-address** keyword is not used, the **debug ephone keepalive** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a sample output of the keepalive status for the Cisco IP phone with MAC address 0030.94C3.E1A8:

```
Router# debug ephone keepalive mac-address 0030.94c3.E1A8

EPHONE keepalive debugging is enabled for phone 0030.94C3.E1A8

1d05h: ephone-1 Set interface FastEthernet0/0 ETHERNET
1d05h: ephone-1[1]:Keepalive socket[1] SEP003094C3E1A8
1d05h: ephone-1 Set interface FastEthernet0/0 ETHERNET
1d05h: ephone-1[1]:Keepalive socket[1] SEP003094C3E1A8
1d05h: Skinny Checking for stale sockets
1d05h: ephone-1 Set interface FastEthernet0/0 ETHERNET
1d05h: ephone-1[1]:Keepalive socket[1] SEP003094C3E1A8
1d05h: ephone-1 Set interface FastEthernet0/0 ETHERNET
1d05h: ephone-1[1]:Keepalive socket[1] SEP003094C3E1A8
1d05h: Skinny active socket list (3/96): 1 2 4
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone mwi</b>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone mwi

To set message waiting indication (MWI) debugging for the Cisco IOS Telephony Service router, use the **debug ephone mwi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone mwi**

**no debug ephone mwi**

**Syntax Description** This command has no arguments or keywords.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Release	Modification
12.2(2)XT	This command was introduced on the following platforms: Cisco 1750, Cisco 1751, Cisco 2600 series and Cisco 3600 series multiservice routers; and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone mwi** command sets message waiting indication debugging for the Cisco IOS Telephony Service router. Since the MWI protocol activity is not specific to any individual Cisco IP phone, setting the MAC address keyword qualifier for this command is not useful.



**Note** Unlike the other related **debug ephone** commands, the **mac-address** keyword does not help debug a particular Cisco IP phone.

**Examples** The following example shows a sample output of the message waiting indication status for the Cisco IOS Telephony Service router:

```
Router# debug ephone mwi
```

## Related Commands

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone pak

To provide voice packet level debugging and to print the contents of one voice packet in every 1024 voice packets, use the **debug ephone pak** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone pak** [**mac-address** *mac-address*]

**no debug ephone pak** [**mac-address** *mac-address*]

Syntax Description	Parameter	Description
	<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
	<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
	12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone pak** command provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.

If the **mac-address** keyword is not used, the **debug ephone pak** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

## Examples

The following example shows a sample output of packet debugging for the Cisco IP phone with MAC address 0030.94c3.8724:

```
Router# debug ephone pak mac-address 0030.94c3.8724

EPHONE packet debugging is enabled for phone 0030.94c3.8724

01:29:14: ***ph_xmit_ephone DN 3 tx_pkts 5770 dest=10.2.1.1 orig len=32
  pakcopy=0 discards 27 ip_enctype 0 0 last discard: unsupported payload type
01:29:14: to_skinny_duration 130210 offset -30 last -40 seq 0 adj 0
01:29:14: IP: 45B8 003C 0866 0000 3F11 3F90 2800 0001 0A02 0101
01:29:14: TTL 63 TOS B8 prec 5
01:29:14: UDP: 07D0 6266 0028 0000
01:29:14: sport 2000 dport 25190 length 40 checksum 0
01:29:14: RTP: 8012 16AF 9170 6409 0E9F 0001
01:29:14: is_rtp:1 is_frfl1:0 vlen:0 delta_t:160 vofr1:0 vofr2:0
scodec:11 rtp_bits:8012 rtp_codec:18 last_bad_payload 19
01:29:14: vencap FAILED
01:29:14: PROCESS SWITCH
01:29:15: %SYS-5-CONFIG_I: Configured from console by console
01:29:34: ***SkinnyPktIp DN 3 10.2.1.1 to 40.0.0.1 pkts 4880 FAST sw
01:29:34: from_skinny_duration 150910
01:29:34: nw 3BBC2A8 addr 3BBC2A4 mac 3BBC2A4 dg 3BBC2C4 dgs 2A
01:29:34: MAC: 1841 0800
01:29:34: IP: 45B8 0046 682E 0000 3E11 E0BD 0A02 0101 2800 0001
01:29:34: TTL 62 TOS B8 prec 5
01:29:34: UDP: 6266 07D0 0032 0000
01:29:34: sport 25190 dport 2000 length 50 checksum 0
01:29:34: RTP: 8012 55FF 0057 8870 3AF4 C394
01:29:34: RTP: rtp_bits 8012 seq 55FF ts 578870 ssrc 3AF4C394
01:29:34: PAYLOAD:
01:29:34: 1409 37C9 54DE 449C 3B42 0446 3AAB 182E
01:29:34: 56BC 5184 58E5 56D3 13BE 44A7 B8C4
01:29:34:
01:29:37: ***ph_xmit_ephone DN 3 tx_pkts 6790 dest=10.2.1.1 orig len=32
  pakcopy=0 discards 31 ip_enctype 0 0 last discard: unsupported payload type
01:29:37: to_skinny_duration 153870 offset -150 last -40 seq 0 adj 0
01:29:37: IP: 45B8 003C 0875 0000 3F11 3F81 2800 0001 0A02 0101
01:29:37: TTL 63 TOS B8 prec 5
01:29:37: UDP: 07D0 6266 0028 0000
01:29:37: sport 2000 dport 25190 length 40 checksum 0
01:29:37: RTP: 8012 1AAF 9173 4769 0E9F 0001
01:29:37: is_rtp:1 is_frfl1:0 vlen:0 delta_t:160 vofr1:0 vofr2:0
```

## Related Commands

Command	Description
<a href="#">debug ephone alarm</a>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<a href="#">debug ephone detail</a>	Sets detail debugging for the Cisco IP phone.
<a href="#">debug ephone error</a>	Sets error debugging for the Cisco IP phone.
<a href="#">debug ephone keepalive</a>	Sets keepalive debugging for the Cisco IP phone.
<a href="#">debug ephone mwi</a>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<a href="#">debug ephone raw</a>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<a href="#">debug ephone register</a>	Sets registration debugging for the Cisco IP phone.
<a href="#">debug ephone state</a>	Sets state debugging for the Cisco IP phone.

Command	Description
<a href="#">debug ephone statistics</a>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone raw

To provide raw low-level protocol debugging display for all Skinny Client Control Protocol messages, use the **debug ephone raw** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone raw** [**mac-address** *mac-address*]

**no debug ephone raw** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone raw** command provides raw low-level protocol debug display for all Skinny Client Control Protocol messages. The debug display provides byte level display of Skinny TCP socket messages.

If the **mac-address** keyword is not used, the **debug ephone raw** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a sample output of raw protocol debugging for the Cisco IP phone with MAC address 0030.94c3.E1A8:

```
Router# debug ephone raw mac-address 0030.94c3.E1A8

EPHONE raw protocol debugging is enabled for phone 0030.94C3.E1A8

1d05h: skinny socket received 4 bytes on socket [1]
0 0 0 0
1d05h:
1d05h: SkinnyMessageID = 0
1d05h: skinny send 4 bytes
4 0 0 0 0 0 0 0 0 1 0 0
1d05h: socket [1] sent 12 bytes OK (incl hdr) for ephone-(1)

1d06h: skinny socket received 4 bytes on socket [1]
0 0 0 0
1d06h:
1d06h: SkinnyMessageID = 0
1d06h: skinny send 4 bytes
4 0 0 0 0 0 0 0 0 1 0 0
1d06h: socket [1] sent 12 bytes OK (incl hdr) for ephone-(1)
```

**Related Commands<sup>1</sup>**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone mwi</b>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone register

To set registration debugging for the Cisco IP phone, use the **debug ephone register** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone register** [**mac-address** *mac-address*]

**no debug ephone register** [**mac-address** *mac-address*]

Syntax Description	mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
	<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
	12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone register** command sets registration debugging for the Cisco IP phones.

If the **mac-address** keyword is not used, the **debug ephone register** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a sample output of registration debugging for the Cisco IP phone with MAC address 0030.94c3.8724:

```
Router# debug ephone register mac-address 0030.94c3.8724

Ephone registration debugging is enabled

1d06h: New Skinny socket accepted [1] (2 active)
1d06h: sin_family 2, sin_port 50778, in_addr 10.1.0.21
1d06h: skinny_add_socket 1 10.1.0.21 50778
1d06h: ephone-(1) [1] StationRegisterMessage (2/3/12) from 10.1.0.21
1d06h: ephone-(1) [1] Register StationIdentifier DeviceName SEP003094C3E1A8
1d06h: ephone-(1) [1] StationIdentifier Instance 1 deviceType 7
1d06h: ephone-1[-1]:stationIpAddr 10.1.0.21
1d06h: ephone-1[-1]:maxStreams 0
1d06h: ephone-(1) Allow any Skinny Server IP address 10.1.0.6
.
.
.
1d06h: ephone-1[1]:RegisterAck sent to ephone 1: keepalive period 30
.
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone mwi</b>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone state

To set state debugging for the Cisco IP phone, use the **debug ephone state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone state** [**mac-address** *mac-address*]

**no debug ephone state** [**mac-address** *mac-address*]

Syntax Description	Parameter	Description
	<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
	<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
	12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone state** command sets state debugging for the Cisco IP phones.

If the **mac-address** keyword is not used, the **debug ephone state** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a sample output of state debugging for the Cisco IP phone with MAC address 0030.94c3.E1A8:

```
Router# debug ephone state mac-address 0030.94c3.E1A8

EPHONE state debugging is enabled for phone 0030.94C3.E1A8

1d06h: ephone-1[1]:OFFHOOK
1d06h: ephone-1[1]:SIEZE on activeline 0
1d06h: ephone-1[1]:SetCallState line 1 DN 1 TsOffHook
1d06h: ephone-1[1]:Skinny-to-Skinny call DN 1 to DN 2 instance 1
1d06h: ephone-1[1]:SetCallState line 1 DN 1 TsRingOut
1d06h: ephone-1[1]:Call Info DN 1 line 1 ref 158 called 5002 calling 5001
1d06h: ephone-1[1]: Jane calling
1d06h: ephone-1[1]: Jill
1d06h: ephone-1[1]:SetCallState line 3 DN 2 TsRingIn
1d06h: ephone-1[1]:Call Info DN 2 line 3 ref 159 called 5002 calling 5001
1d06h: ephone-1[1]: Jane calling
1d06h: ephone-1[1]: Jill
1d06h: ephone-1[1]:SetCallState line 3 DN 2 TsCallRemoteMultiline
1d06h: ephone-1[1]:SetCallState line 1 DN 1 TsConnected
1d06h: ephone-1[1]:OpenReceive DN 1 codec 4:G711Ulaw64k duration 10 ms bytes 80
1d06h: ephone-1[1]:OpenReceiveChannelAck 1.2.172.21 port=24010
1d06h: ephone-1[1]:StartMedia 1.2.172.22 port=24612
1d06h: DN 1 codec 4:G711Ulaw64k duration 10 ms bytes 80
1d06h: ephone-1[1]:CloseReceive
1d06h: ephone-1[1]:StopMedia
1d06h: ephone-1[1]:SetCallState line 3 DN 2 TsOnHook
1d06h: ephone-1[1]:SetCallState line 1 DN 1 TsOnHook
1d06h: ephone-1[1]:SpeakerPhoneOnHook
1d06h: ephone-1[1]:ONHOOK
1d06h: ephone-1[1]:SpeakerPhoneOnHook
1d06h: SkinnyReportDnState DN 1 ONHOOK
```

**Related Commands**

Command	Description
<a href="#">debug ephone alarm</a>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<a href="#">debug ephone detail</a>	Sets detail debugging for the Cisco IP phone.
<a href="#">debug ephone error</a>	Sets error debugging for the Cisco IP phone.
<a href="#">debug ephone keepalive</a>	Sets keepalive debugging for the Cisco IP phone.
<a href="#">debug ephone mwi</a>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<a href="#">debug ephone pak</a>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<a href="#">debug ephone raw</a>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<a href="#">debug ephone register</a>	Sets registration debugging for the Cisco IP phone.
<a href="#">debug ephone statistics</a>	Sets statistics debugging for the Cisco IP phone.
<a href="#">show debugging</a>	Displays information about the types of debugging that are enabled for your router.

# debug ephone statistics

To set call statistics debugging for the Cisco IP phone, use the **debug ephone statistics** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone statistics** [**mac-address** *mac-address*]

**no debug ephone statistics** [**mac-address** *mac-address*]

Syntax Description	mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
	<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
	12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on the Cisco 1760 routers.

**Usage Guidelines** The **debug ephone statistics** command provides a debug monitor display of the periodic messages from the Cisco IP phone to the router. These include transmit-and-receive packet counts and an estimate of drop packets. The call statistics can also be displayed for live calls using the **show ephone** command.

If the **mac-address** keyword is not used, the **debug ephone statistics** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a sample output of statistics debugging for the Cisco IP phone with MAC address 0030.94C3.E1A8:

```
Router# debug ephone statistics mac-address 0030.94C3.E1A8

EPHONE statistics debugging is enabled for phone 0030.94C3.E1A8

1d06h: Clear Call Stats for DN 1 call ref 162
1d06h: Clear Call Stats for DN 1 call ref 162
1d06h: Clear Call Stats for DN 1 call ref 162
1d06h: Clear Call Stats for DN 2 call ref 163
1d06h: ephone-1[1]:GetCallStats line 1 ref 162 DN 1: 5001
1d06h: ephone-1[1]:Call Stats for line 1 DN 1 5001 ref 162
1d06h: ephone-1[1]:TX Pkts 0 bytes 0 RX Pkts 0 bytes 0
1d06h: ephone-1[1]:Pkts lost 4504384 jitter 0 latency 0
1d06h: ephone-1[1]:Src 0.0.0.0 0 Dst 0.0.0.0 0 bytes 80 vad 0 G711Ulaw64k
1d06h: ephone-1[1]:GetCallStats line 1 ref 162 DN 1: 5001
1d06h: STATS: DN 1 Packets Sent 0
1d06h: STATS: DN 2 Packets Sent 0
1d06h: ephone-1[1]:Call Stats found DN -1 from Call Ref 162
1d06h: ephone-1[1]:Call Stats for line 0 DN -1 5001 ref 162
1d06h: ephone-1[1]:TX Pkts 275 bytes 25300 RX Pkts 275 bytes 25300
1d06h: ephone-1[1]:Pkts lost 0 jitter 0 latency 0
```

**Related Commands**

Command	Description
<a href="#">debug ephone alarm</a>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<a href="#">debug ephone detail</a>	Sets detail debugging for the Cisco IP phone.
<a href="#">debug ephone error</a>	Sets error debugging for the Cisco IP phone.
<a href="#">debug ephone keepalive</a>	Sets keepalive debugging for the Cisco IP phone.
<a href="#">debug ephone mwi</a>	Sets message waiting indication (MWI) debugging for the Cisco IP phone.
<a href="#">debug ephone pak</a>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<a href="#">debug ephone raw</a>	Provides raw low-level protocol debugging display for all Skinny Client Control Protocol messages.
<a href="#">debug ephone register</a>	Sets registration debugging for the Cisco IP phone.
<a href="#">debug ephone state</a>	Sets state debugging for the Cisco IP phone.
<a href="#">show debugging</a>	Displays information about the types of debugging that are enabled for your router.

# debug errors

To display errors, use the **debug errors** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug errors**

**no debug errors**

---

**Syntax Description** This command has no arguments or keywords.

---

**Command Modes** Privileged EXEC

---

**Examples** The following is sample output from the **debug errors** command:

```
Router# debug errors  
  
(2/0): Encapsulation error, link=7, host=836CA86D.  
(4/0): VCD#7 failed to echo OAM. 4 tries
```

The first line of output indicates that a packet was routed to the interface, but no static map was set up to route that packet to the proper virtual circuit.

The second line of output shows that an OAM F5 (virtual circuit) cell error occurred.

# debug events

To display events, use the **debug events** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug events**

**no debug events**

---

**Syntax Description** This command has no arguments or keywords.

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command displays events that occur on the interface processor and is useful for diagnosing problems in a network. It provides an overall picture of the stability of the network. In a stable network, the **debug events** command does not return any information. If the command generates numerous messages, the messages can indicate the possible source of problems.

When configuring or making changes to a router or interface for, enable the **debug events** command. Doing so alerts you to the progress of the changes or to any errors that might result. Also use this command periodically when you suspect network problems.

---

**Examples** The following is sample output from the **debug events** command:

```
Router# debug events

RESET(4/0): PLIM type is 1, Rate is 100Mbps
aip_disable(4/0): state=1
config(4/0)
aip_love_note(4/0): asr=0x201
aip_enable(4/0)
aip_love_note(4/0): asr=0x4000
aip_enable(4/0): restarting VCs: 7
aip_setup_vc(4/0): vc:1 vpi:1 vci:1
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:2 vpi:2 vci:2
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:3 vpi:3 vci:3
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:4 vpi:4 vci:4
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:6 vpi:6 vci:6
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:7 vpi:7 vci:7
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:11 vpi:11 vci:11
aip_love_note(4/0): asr=0x200
```

Table 61 describes the significant fields in the display.

**Table 61** *debug events Command Field Descriptions*

Field	Description
PLIM type	Indicates the interface rate in Mbps. Possible values are: <ul style="list-style-type: none"> <li>• 1 = TAXI(4B5B) 100 Mbps</li> <li>• 2 = SONET 155 Mbps</li> <li>• 3 = E3 34 Mbps</li> </ul>
state	Indicates current state of the AIP. Possible values are: <ul style="list-style-type: none"> <li>• 1 = An ENABLE will be issued soon.</li> <li>• 0 = The AIP will remain shut down.</li> </ul>
asr	Defines a bitmask, which indicates actions or completions to commands. Valid bitmask values are: <ul style="list-style-type: none"> <li>• 0x0800 = AIP crashed, reload may be required.</li> <li>• 0x0400 = AIP detected a carrier state change.</li> <li>• 0x0n00 = Command completion status. Command completion status codes are:               <ul style="list-style-type: none"> <li>– n = 8 Invalid PLIM detected</li> <li>– n = 4 Command failed</li> <li>– n = 2 Command completed successfully</li> <li>– n = 1 CONFIG request failed</li> <li>– n = 0 Invalid value</li> </ul> </li> </ul>

The following line indicates that the AIP was reset. The PLIM detected was 1, so the maximum rate is set to 100 Mbps.

```
RESET(4/0): PLIM type is 1, Rate is 100Mbps
```

The following line indicates that the AIP was given a **shutdown** command, but the current configuration indicates that the AIP should be up:

```
aip_disable(4/0): state=1
```

The following line indicates that a configuration command has been completed by the AIP:

```
aip_love_note(4/0): asr=0x201
```

The following line indicates that the AIP was given a **no shutdown** command to take it out of the shutdown state:

```
aip_enable(4/0)
```

The following line indicates that the AIP detected a carrier state change. It does not indicate that the carrier is down or up, only that it has changed.

```
aip_love_note(4/0): asr=0x4000
```

The following line of output indicates that the AIP enable function is restarting all PVCs automatically:

```
aip_enable(4/0): restarting VCs: 7
```

The following lines of output indicate that PVC 1 was set up and a successful completion code was returned:

```
aip_setup_vc(4/0): vc:1 vpi:1 vci:1  
aip_love_note(4/0): asr=0x200
```

# debug fax relay t30

To display debug messages for T.30 real-time fax, use the **debug fax relay t30** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug fax relay t30** {**all** | **calling-number** *string* | **called-number** *string*}

**no debug fax relay t30**

Syntax Description		
	<b>all</b>	Enables debugging for all incoming and outgoing calls.
	<b>calling-number</b>	Enables debugging for incoming numbers that begin with a specified string of digits.
	<b>called-number</b>	Enables debugging for outgoing numbers that begin with a specified string of digits.
	<i>string</i>	Digits that specify the incoming or outgoing number.

**Defaults** No default behavior or values.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.2(2)XB1	The <b>debug fax relay t30</b> command was introduced on Cisco AS5350, Cisco AS5400, and Cisco AS5850 access servers.
	12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and support was added for the Cisco AS5350, Cisco AS5400, and Cisco AS5850 access servers.

**Usage Guidelines** The incoming or outgoing numbers must be a valid E.164 destination. The period symbol (.) as a wild card should not be used. Instead of a wild card, leave the space blank to indicate that any numbers can be valid.

There are no limits to the number of debug entries. The number entered generates a match if the calling or called number matches up to the final number of the debug entry. For example, the 408555 entry would match 408555, 4085551, 4085551212, or any other number starting with 408555.

**Examples** The following command enables debugging for any incoming calls that start with 408555:

```
Router# debug fax relay t30 calling-number 408555
```

```
Debugging fax relay t30 from 408555
```

The following command enables debugging for any calls received to a number starting with 555-1212:

```
Router# debug fax relay t30 called-number 4155551212
```

```
Debugging fax relay t30 to 4155551212
```

The following command displays all debug entries:

```
Router# debug fax relay t30 all
```

```
Debugging fax relay t30 from 408555
```

```
Debugging fax relay t30 to 4155551212
```

# debug fddi smt-packets

To display information about Station Management (SMT) frames received by the router, use the **debug fddi smt-packets** command in privileged EXEC mode. The **no** form of this command disables debugging output.

**debug fddi smt-packets**

**no debug fddi smt-packets**

**Syntax Description** This command has no arguments or keywords.

**Command Modes** Privileged EXEC

**Examples** The following is sample output from the **debug fddi smt-packets** command. In this example, an SMT frame has been output by FDDI 1/0. The SMT frame is a next station addressing (NSA) neighbor information frame (NIF) request frame with the parameters as shown.

```
Router# debug fddi smt-packets

SMT O: Fddi1/0, FC=NSA, DA=ffff.ffff.ffff, SA=00c0.eeee.be04,
class=NIF, type=Request, vers=1, station_id=00c0.eeee.be04, len=40
- code 1, len 8 -- 000000016850043F
- code 2, len 4 -- 00010200
- code 3, len 4 -- 00003100
- code 200B, len 8 -- 0000000100000000
```

[Table 62](#) describes the significant fields shown in the display.

**Table 62** *debug fddi smt-packets Command Field Descriptions*

Field	Description
SMT O	SMT frame was sent from FDDI interface 1/0. Also, SMT I indicates that an SMT frame was received on the FDDI interface 1/0.
Fddi1/0	Interface associated with the frame.
FC	Frame control byte in the MAC header.
DA, SA	Destination and source addresses in FDDI form.
class	Frame class. Values can be echo frame (ECF), neighbor information frame (NIF), parameter management frame (PMF), request denied frame (RDF), status information frame (SIF), and status report frame (SRF).
type	Frame type. Values can be Request, Response, and Announce.
vers	Version identification. Values can be 1 or 2.
station_id	Station identification.

**Table 62** *debug fddi smt-packets Command Field Descriptions (continued)*

Field	Description
len	Packet size.
code 1, len 8 -- 000000016850043F	Parameter type X'0001—upstream neighbor address (UNA), parameter length in bytes, and parameter value. SMT parameters are described in the SMT specification ANSI X3T9.