

# debug cable env

To display information about the Cisco uBR7246 universal broadband router physical environment, including internal temperature, midplane voltages, fan performance, and power supply voltages, use the **debug cable env** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable env**

**no debug cable env**

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

**Command Modes** Privileged EXEC

**Usage Guidelines** This command is used to debug the sensor circuitry used to measure internal temperature, midplane voltages, fan performance, and power supply voltages on the Cisco uBR7246 console.

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

```
Router# debug cable env

ENVM: ps id=0xFF0, v=0x2050, r=0xC0AB, pstype=1
ENVM: ps id=0x2FD0, v=0x2050, r=0x24201, pstype=27
ENVM: Sensor 0: a2dref=131, a2dact=31, vref=12219, vact=1552
      Alpha=8990, temp=27
```

[Table 30](#) describes the significant fields shown in the display.

**Table 30** *debug cable env Field Descriptions*

Field	Description
ps id	Power supply raw voltage reading.
pstype	Power supply type determined from the ps id, v, and r values. The Cisco uBR7246 universal broadband router contains dual power supplies, so ID information for two types is usually printed.
Sensor	Sensor number.
a2dref	Analog-to-digital converter reference reading.
a2dact	Analog-to-digital converter actual (measured reading).
vref	Reference voltage.
vact	Actual voltage.
Alpha	Raw temperature reading.
temp	Temperature corresponding to Alpha.

# debug cable err

To display errors that occur in the cable MAC protocols, use the **debug cable err** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable err**

**no debug cable err**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command is used to display unexpected Data-over-Cable Service Interface Specifications (DOCSIS) MAC protocol messages. When the Cisco uBR7246 universal broadband router does not expect to receive a specific MAC message, an error message and hexadecimal dump are printed. Other miscellaneous error conditions may result in output.

---

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

```
Router# debug cable err

This is a UCD Message
This is a MAP Message
This is a RNG_RSP Message
This is a REG_RSP Message
This is a UCC_REQ Message
This is a BPKM_RSP Message
This is a TRI_TCD Message
This is a TRI_TSI Message
This is a unrecognized MCNS message

ERROR:#####TICKS PER MSLOT NOT POWER OF 2####
```

# debug cable freqhop

To display debug messages for frequency hopping, use the **debug cable freqhop** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable freqhop**

**no debug cable freqhop**

---

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

---

**Defaults** Debugging for frequency hopping is not enabled.

---

**Command Modes** Privileged EXEC

---

Command History	Release	Modification
	12.0(4)XI	This command was introduced.

---



---

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

```
Router# debug cable freqhop
CMTS freqhop debugging is on
```

---

Related Commands	Command	Description
	<b>debug cable hw-spectrum</b>	Displays debug information about spectrum management (frequency agility).

---

# debug cable hw-spectrum

To display debug messages for spectrum management (frequency agility), use the **debug cable hw-spectrum** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable hw-spectrum**

**no debug cable hw-spectrum**

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

**Defaults** Debugging for spectrum management is not enabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0	This command was introduced as <b>debug cable specmgmt</b> .
	12.0(4)XI	This command was renamed as <b>debug cable hw-spectrum</b> .

**Examples** The following is sample output from the **debug cable hw-spectrum** command:

```
Router# debug cable hw-spectrum
CMTS specmgmt debugging is on
```

# debug cable interface

To perform debugging on a specified interface, use the **debug cable interface** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable interface** *interface-type interface-number* [**mac-address** *address* | **mask** | **verbose**]

**no debug cable interface** *interface-type interface-number* **mac-address** *address*

## Syntax Description

<i>interface-type</i>	<i>interface-number</i>	Specifies the cable interface to be debugged. A space is not required between the values.
<b>mac-address</b>		(Optional) Specifies that debugging is to be done on a specified MAC address.
	<i>address</i>	(Optional) Specifies the MAC address of the interface.
<b>mask</b>		(Optional) Specifies the MAC address validation address.
<b>verbose</b>		(Optional) Displays detailed debug information.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.0(6)T	This command was introduced.

## Usage Guidelines

You can repeat this **debug** command for other interfaces. Each time you specify a different cable interface or MAC address, debugging is turned on for this cable interface or MAC address.

If you enter two **debug** commands with the same interface or MAC address, but with different mask or verbose keywords, the router treats both commands as the same. In this case, the latest debug information supersedes the previous debugging information.

## Examples

The following example demonstrates how to enable debugging on interface c3/0:

```
Router# debug cable interface c3/0
```

The following example demonstrates how to enable detailed debugging on interface c3/0:

```
Router# debug cable interface c3/0 verbose
```

The following example demonstrates how to enable debugging on interface c3/0 for all traffic coming from modems with MAC addresses 0010.00xx.xxxx:

```
Router# debug cable interface c3/0 mac-address 0010.0000.0000 ffff.ff00.0000
```

## Related Commands

Command	Description
<a href="#">debug cable mac-address</a>	Enables debugging on traffic from modems with the specified MAC address or MAC address range.

# debug cable keyman

To activate debugging of traffic encryption key (TEK) and key-encrypting key (KEK) baseline privacy key activity, use the **debug cable keyman** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable keyman**

**no debug cable keyman**

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

**Command Modes** Privileged EXEC

**Usage Guidelines** This command activates debugging of the TEK and KEK baseline privacy key activity. When this command is activated, all activity related to KEK and TEK keys will be displayed on the Cisco uBR7246 console. This command is used to display encryption key management debugging output.

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

```
Router# debug cable keyman

Read Verify DES failed with SID %2x
  Verify key failed with SID %2x : setvalue = %llx, readback = %llx
  Verify iv failed with SID %2x : setvalue = %llx, readback = %llx
Next TEK lifetime check is set to %u seconds.
  Next Multicast TEK lifetime check is set to 1 seconds

[UCAST_TEK] :", idbp->hw_namestring);
  show_sid_key_chain(ds, &ds->mcast_sid_key_list_hdr);

[MCAST_TEK] :", idbp->hw_namestring);
  buginf("\nSID : %4x\t", sidkey->sid);
  buginf("seq : %2x\t current : %2x\n", sidkey->key_seq_num,
  sidkey->current_key_num);
  buginf(" Status[0] : %x\tDES IV[0] : %llx\tKey Life[0]: %u sec\n",
  sidkey->key_status[0], sidkey->des_key[0].iv,
  compute_remain_lifetime(&sidkey->des_key[0]));

  buginf(" Status[1] : %x\tDES IV[1] : %llx\tKey Life[1]: %u sec\n",
  sidkey->key_status[1], sidkey->des_key2131.iv,
  compute_remain_lifetime(&sidkey->des_key[1]));
```

# debug cable mac

To display MAC-layer information for the specified cable modem, use the **debug cable mac** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable mac**

**no debug cable mac**

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

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3 NA	This command was introduced.

**Usage Guidelines** Do not use this command if you have a large number of modems on your network. The Cisco uBR7246 universal broadband router will become flooded with console printouts.

**Examples** The following example shows the return for the MAC layer:

```
Router# debug cable mac

19:46:27: Ranging Modem with Sid 1 on i/f : Cable6/0/U0

19:46:27: Got a ranging request
19:46:27: SID value is 1 on Interface Cable6/0/U0
19:46:27: CM mac address 00:E0:1E:B2:BB:07
19:46:27: Timing offset is 0
19:46:27: Power value is FE0, or 0 dB
19:46:27: Freq Error = 0, Freq offset is 0
19:46:27: Ranging has been successful for SID 1 on Interface Cable6/0/U0

19:46:29: Ranging Modem with Sid 2 on i/f : Cable6/0/U0
19:46:29: Got a ranging request
19:46:29: SID value is 2 on Interface Cable6/0/U0
19:46:29: CM mac address 00:E0:1E:B2:BB:8F
19:46:29: Timing offset is 1
19:46:29: Power value is 1350, or 0 dB
19:46:29: Freq Error = 0, Freq offset is 0
19:46:29: Ranging has been successful for SID 2 on Interface Cable6/0/U0

19:46:32: Ranging Modem with Sid 3 on i/f : Cable6/0/U0

19:46:32: Got a ranging request
19:46:32: SID value is 3 on Interface Cable6/0/U0
19:46:32: CM mac address 00:E0:1E:B2:BB:B1
19:46:32: Timing offset is FFFFFFFF
19:46:32: Power value is 1890, or -1 dB
19:46:32: Freq Error = 0, Freq offset is 0
```

19:46:32: Ranging has been successful for SID 3 on Interface Cable6/0/U0

19:46:34: Ranging Modem with Sid 5 on i/f : Cable6/0/U0

Table 31 describes the significant fields shown in the display.

**Table 31** *debug cable mac Field Descriptions*

Field	Description
SID value is....	Reports the service ID of the modem. The range is from 1 through 891. The information on this line should agree with the first line of the return (that is, Ranging Modem with Sid...).
CM mac address....	MAC address of the specified cable modem.
Timing offset is....	Time by which to offset the frame transmission upstream so the frame arrives at the expected minislot time at the cable modem termination system (CMTS).
Power value is FE0, or 0 dB	Raw value derived from the 3137 Broadcom chip. Alternately, the decibel value specifies the relative change in the transmission power level that the cable modem needs to make so transmissions arrive at the CMTS at the desired power level. This desired power level is usually 0, but you can use the CLI to change it via the <b>cable power-level</b> command.
Freq Error = ....	Raw value derived from the 3137 Broadcom chip.
Freq offset is ....	Specifies the relative change in the transmission frequency that the cable modem will make to match the CMTS.

#### Related Commands

Command	Description
<b>show controllers cable</b>	Displays interface controller information for the specified slot.

# debug cable mac-address

To enable debugging for a specified MAC address, use the **debug cable mac-address** command in privileged EXEC mode. To turn off debugging output, use the **no** form of this command.

**debug cable mac-address** *address* [**mask** | **verbose**]

**no debug cable mac-address** *address*

Syntax Description		
	<i>address</i>	Specifies the MAC address of the interface.
	<b>mask</b>	(Optional) Specifies the MAC address validation address.
	<b>verbose</b>	(Optional) Displays detailed debug information.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(6)T	This command was introduced.

**Usage Guidelines** You can repeat this **debug** command for other MAC addresses. Each time you specify a different MAC address, debugging is turned on for this MAC address.

If you enter two **debug** commands with the same MAC address, but with different mask or verbose keywords, the router treats both commands as the same. In this case, the latest debug information supersedes the previous debugging information.

**Examples** The following example demonstrates how to enable debugging for all traffic coming from all interfaces of modems with the MAC address 0010.00xx.xxxx:

```
Router# debug cable mac-address 0010.0000 ffff.ff00.000
```

Related Commands	Command	Description
	<a href="#">debug cable interface</a>	Enables debugging on the cable interface specified.

# debug cable map

To display map debugging messages, use the **debug cable map** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable map**

**no debug cable map**

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

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3 NA	This command was introduced.

**Examples** The following example displays all the map messages with and without data grants:

```
Router# debug cable map

19:41:53: On interface Cable6/0, sent 5000 MAPs, 1321 MAPs had grant(s)Long Grants
13256993, Total Short Grants 223
A sample Map without any data grant
----- MAP MSG -----
us_ch_id: 1   ucd_count: 5   num_elems: 9   reserved: 0
Alloc Start Time: 33792           Ack Time: 33618
Rng_bkoff_start: 0   Rng_bkoff_end: 2
Data_bkoff_start: 1   Data_bkoff_end: 3:
sid:16383   iuc:1   mslot_offset:0
sid:0   iuc:7   mslot_offset:40
A sample Map with data grant(s)
----- MAP MSG -----
us_ch_id: 1   ucd_count: 5   num_elems: 7   reserved: 0
Alloc Start Time: 33712           Ack Time: 33578
Rng_bkoff_start: 0   Rng_bkoff_end: 2
Data_bkoff_start: 1   Data_bkoff_end: 3
sid:2   iuc:6   mslot_offset:0
sid:16383   iuc:1   mslot_offset:16
sid:0   iuc:7   mslot_offset:40
```

[Table 32](#) describes the significant fields shown in the display.

**Table 32** *debug cable map Field Descriptions*

Field	Description
sent 5000 MAPs	Total number of maps sent.
MAPs had grant(s) Long Grants	Total number of grants considered long sized by the cable modem termination system (CMTS).
Total Short Grants	Total number of grants considered short sized by the CMTS.

Table 32 *debug cable map Field Descriptions (continued)*

Field	Description
us_ch_id	Identifies the upstream channel ID for this message.
ucd_count	Number of upstream channel descriptors (UCDs).
num_elems	Number of information elements in the map.
reserved	Reserved for alignment.
Alloc Start Time	Start time from CMTS initialization (in minislots) for assignments in this map.
Ack Time	Latest time from CMTS initialization (in minislots) processed in upstream. The cable modems use this time for collision detection.
Rng_bkoff_start	Initial backoff window for initial ranging contention, expressed as a power of 2. Valid values are from 0 to 15.
Rng_bkoff_end	Final backoff window for initial ranging contention, expressed as a power of 2. Valid values are from 0 to 15.
Data_bkoff_start	Initial backoff window for contention data and requests, expressed as a power of 2. Valid values are from 0 to 15.
Data_bkoff_end	Final backoff window for contention data and requests, expressed as a power of 2. Valid values are from 0 to 15.
sid	Service ID.
iuc	Interval usage code (IUC) value.
mslot_offset	Minislot offset.

## Related Commands

Command	Description
<b>show controllers cable</b>	Displays interface controller information for the specified slot.

# debug cable-modem bpkm

To debug baseline privacy information, use the **debug cable-modem bpkm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug cable-modem bpkm {errors | events | packets}
```

```
no debug cable-modem bpkm {errors | events | packets}
```

Syntax Description	errors	Provides debugging information about Cisco uBR900 series privacy errors.
	events	Provides debugging information about events related to cable baseline privacy.
	packets	Provides debugging information about baseline privacy packets.

Command Modes	Privileged EXEC
---------------	-----------------

Command History	Release	Modification
	11.3 NA	This command was introduced for the Cisco uBR900 series cable access router.

**Usage Guidelines** Baseline privacy key management exchanges take place only when both the Cisco uBR900 series and the cable modem termination system (CMTS) are running code images that support baseline privacy, and the privacy class of service is enabled via the configuration file that is downloaded to the cable modem. Baseline privacy code images for the Cisco uBR900 series contain “k1” in the code image name.

**Examples** The following is sample output from the **debug cable-modem bpkm errors** command when the headend does not have privacy enabled:

```
Router# debug cable-modem bpkm errors

cm_bpkm_fsm(): machine: KEK, event/state: EVENT_4_TIMEOUT/STATE_B_AUTH_WAIT, new state:
STATE_B_AUTH_WAIT

cm_bpkm_fsm(): machine: KEK, event/state: EVENT_4_TIMEOUT/STATE_B_AUTH_WAIT, new state:
STATE_B_AUTH_WAIT

%LINEPROTO-5-UPDOWN: Line protocol on Interface cable-modem0, changed state to down
cm_bpkm_fsm(): machine: KEK, event/state: EVENT_1_PROVISIONED/STATE_A_START, new state:
STATE_B_AUTH_WAIT

%LINEPROTO-5-UPDOWN: Line protocol on Interface cable-modem0, changed state to up
```

Related Commands	Command	Description
	<b>debug cable-modem bridge</b>	Displays bridge filter processing information.
	<b>debug cable-modem error</b>	Enables debugging messages for the cable interface driver.
	<b>debug cable-modem interrupts</b>	Debugs cable modem interrupts.
	<b>debug cable-modem mac</b>	Troubleshoots the MAC layer for cable modems.
	<b>debug cable-modem map</b>	Displays the timing from MAP messages to synchronize messages and the timing between MAP messages on a Cisco uBR900 series cable access router.

# debug cable-modem bridge

To debug bridge filter processing information, use the **debug cable-modem bridge** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable-modem bridge**

**no debug cable-modem bridge**

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

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3 NA	This command was introduced for the Cisco uBR900 series cable access router.

**Usage Guidelines** When the interface is down, all bridge table entries learned on the Ethernet interface are set to discard because traffic is not bridged until the cable interface has completed initialization. After the interface (the line protocol) is completely up, bridge table entries learned on the Ethernet interface program the cable MAC data filters. The cable MAC hardware filters out any received packets whose addresses are not in the filters. In this way, the cable interface only receives packets addressed to its own MAC address or an address it has learned on the Ethernet interface.

**Examples** The following example is sample output from the **debug cable-modem bridge** command:

```
Router# debug cable-modem bridge

%LINEPROTO-5-UPDOWN: Line protocol on Interface cable-modem0, changed state to downshut
cm_tbridge_add_entry(): MAC not initialized, discarding entry: 00e0.fe7a.186fno shut
cm_tbridge_add_entry(): MAC not initialized, discarding entry: 00e0.fe7a.186f
%LINEPROTO-5-UPDOWN: Line protocol on Interface cable-modem0, changed state to up
cm_tbridge_add_entry(): Adding entry 00e0.fe7a.186f to filter 2
```

Related Commands	Command	Description
	<a href="#">debug cable-modem error</a>	Enables debugging messages.
	<a href="#">debug cable-modem interrupts</a>	Debugs cable modem interrupts.
	<a href="#">debug cable-modem mac</a>	Troubleshoots the MAC layer for cable modems.
	<a href="#">debug cable-modem map</a>	Displays the timing from MAP messages to synchronize messages and the timing between MAP messages on a Cisco uBR900 series cable access router.

# debug cable-modem error

To enable debugging messages for the cable interface driver, use the **debug cable-modem error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable-modem error**

**no debug cable-modem error**

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

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3 NA	This command was introduced.

**Usage Guidelines** This command displays detailed output about the sanity checking of received frame formats, the acquisition of downstream QAM/forward error correction (FEC) lock, the receipt or nonreceipt of SYNC messages from the cable modem termination system (CMTS), reception errors, and bandwidth request failures.

**Examples** The following is sample output from the **debug cable-modem error** command:

```
Router# debug cable-modem error

*Mar 7 20:16:29: AcquireSync(): Update rate is 100 Hz
*Mar 7 20:16:30: 1st Sync acquired after 1100 ms.
*Mar 7 20:16:30: Recovery loop is locked (7/9)
*Mar 7 20:16:30: 2nd Sync acquired after 100 ms.
*Mar 7 20:16:30: Recovery loop is locked (10/15)
```

Related Commands	Command	Description
	<a href="#">debug cable-modem bridge</a>	Displays bridge filter processing information.
	<a href="#">debug cable-modem interrupts</a>	Debugs cable modem interrupts.
	<a href="#">debug cable-modem mac</a>	Troubleshoots the MAC layer for cable modems.
	<a href="#">debug cable-modem map</a>	Displays the timing from MAP messages to synchronize messages and the timing between MAP messages on a Cisco uBR900 series cable access router.

# debug cable-modem interrupts

To debug cable modem interrupts, use the **debug cable-modem interrupts** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable-modem interrupts**

**no debug cable-modem interrupts**

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

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3 NA	This command was introduced for the Cisco uBR900 series.

**Examples** The following is sample output from the **debug cable-modem interrupts** command for Cisco uBR900 series interrupts:

```
Router# debug cable-modem interrupts

*** BCM3300_rx_mac_msg_interrupt ***
*** BCM3300_rx_mac_msg_interrupt ***
### BCM3300_tx_interrupt ###
*** BCM3300_rx_mac_msg_interrupt ***
### BCM3300_tx_interrupt ###
*** BCM3300_rx_mac_msg_interrupt ***
### BCM3300_tx_interrupt ###
### BCM3300_tx_interrupt ###
### BCM3300_tx_interrupt ###
### BCM3300_tx_interrupt ###
```

Related Commands	Command	Description
	<a href="#">debug cable-modem bridge</a>	Displays bridge filter processing information.
	<a href="#">debug cable-modem error</a>	Enables debugging messages for the cable interface driver.
	<a href="#">debug cable-modem mac</a>	Troubleshoots the MAC layer for cable modems.
	<a href="#">debug cable-modem map</a>	Displays the timing from MAP messages to synchronize messages and the timing between MAP messages on a Cisco uBR900 series cable access router.

# debug cable-modem mac

To troubleshoot the MAC layer for cable modems, use the **debug cable-modem mac** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable-modem mac** {log [verbose] | messages}

**no debug cable-modem mac** {log [verbose] | messages}

Syntax Description	log	Displays the real-time MAC log.
	<b>verbose</b>	(Optional) Displays periodic MAC-layer events, such as ranging.
	<b>messages</b>	Displays MAC layer management messages.

Command Modes	Privileged EXEC
---------------	-----------------

Command History	Release	Modification
	11.3 NA	This command was introduced for the Cisco uBR900 series.

**Usage Guidelines**

Of all the available **debug cable-modem** commands, the most useful is **debug cable-modem mac log**. MAC log messages are written to a circular log file even when debugging is not turned on. These messages include time stamps, events, and information pertinent to these events. Enter the **debug cable-modem mac log** command to view MAC log messages. If you want to view this information without entering debug mode, enter the **show controllers cable-modem number mac log** command. The same information is displayed by both commands.

If the Cisco uBR900 series interface fails to come up or resets periodically, the MAC log will show what happened. For example, if an address is not obtained from the Dynamic Host Configuration Protocol (DHCP) server, an error is logged, initialization starts over, and the Cisco uBR900 series cable access server router scans for a downstream frequency. The **debug cable-modem mac log** command displays the log from the oldest to the newest entry.

After initial ranging is successful (dhcp\_state has been reached), further RNG-REQ/RNG-RSP messages and watchdog timer entries are suppressed from output unless the **verbose** keyword is used. Note that CMAC\_LOG\_WATCHDOG\_TIMER entries while in the maintenance\_state are normal when the **verbose** keyword is used.

**Examples**

The following example is sample output from the **debug cable-modem mac log** command. The fields of the output are the time since bootup, the log message, and in some cases a parameter that gives more detail about the log entry.

```
Router# debug cable-modem mac log

*Mar  7 01:42:59: 528302.040 CMAC_LOG_LINK_DOWN
*Mar  7 01:42:59: 528302.042 CMAC_LOG_RESET_FROM_DRIVER
*Mar  7 01:42:59: 528302.044 CMAC_LOG_STATE_CHANGE
wait_for_link_up_state
```

```

*Mar 7 01:42:59: 528302.046 CMAC_LOG_DRIVER_INIT_IDB_SHUTDOWN      0x08098D02
*Mar 7 01:42:59: 528302.048 CMAC_LOG_LINK_DOWN
*Mar 7 01:43:05: 528308.428 CMAC_LOG_DRIVER_INIT_IDB_RESET      0x08098E5E
*Mar 7 01:43:05: 528308.432 CMAC_LOG_LINK_DOWN
*Mar 7 01:43:05: 528308.434 CMAC_LOG_LINK_UP
*Mar 7 01:43:05: 528308.436 CMAC_LOG_STATE_CHANGE
ds_channel_scanning_state
*Mar 7 01:43:05: 528308.440 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
88/453000000/855000000/6000000
*Mar 7 01:43:05: 528308.444 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
89/930000000/105000000/6000000
*Mar 7 01:43:05: 528308.448 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
90/111250000/117250000/6000000
*Mar 7 01:43:05: 528308.452 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
91/231012500/327012500/6000000
*Mar 7 01:43:05: 528308.456 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
92/333015000/333015000/6000000
*Mar 7 01:43:05: 528308.460 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
93/339012500/399012500/6000000
*Mar 7 01:43:05: 528308.462 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
94/405000000/447000000/6000000
*Mar 7 01:43:05: 528308.466 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
95/123015000/129015000/6000000
*Mar 7 01:43:05: 528308.470 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
96/135012500/135012500/6000000
*Mar 7 01:43:05: 528308.474 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
97/141000000/171000000/6000000
*Mar 7 01:43:05: 528308.478 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
98/219000000/225000000/6000000
*Mar 7 01:43:05: 528308.482 CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND
99/177000000/213000000/6000000
*Mar 7 01:43:05: 528308.486 CMAC_LOG_WILL_SEARCH_SAVED_DS_FREQUENCY 663000000
*Mar 7 01:43:05: 528308.488 CMAC_LOG_WILL_SEARCH_USER_DS_FREQUENCY 663000000
*Mar 7 01:43:07: 528310.292 CMAC_LOG_DS_64QAM_LOCK_ACQUIRED      663000000
.
528383.992 CMAC_LOG_STATE_CHANGE                                registration_state
528384.044 CMAC_LOG_REG_REQ_MSG_QUEUED
528384.050 CMAC_LOG_REG_REQ_TRANSMITTED
528384.052 CMAC_LOG_REG_RSP_MSG_RCVD
528384.078 CMAC_LOG_COS_ASSIGNED_SID                          1/4
528384.102 CMAC_LOG_RNG_REQ_QUEUED                            4
528384.102 CMAC_LOG_REGISTRATION_OK
528384.102 CMAC_LOG_STATE_CHANGE                                establish_privacy_state
528384.102 CMAC_LOG_STATE_CHANGE                                maintenance_state
528388.444 CMAC_LOG_RNG_REQ_TRANSMITTED
528388.444 CMAC_LOG_RNG_RSP_MSG_RCVD
528398.514 CMAC_LOG_RNG_REQ_TRANSMITTED
528398.516 CMAC_LOG_RNG_RSP_MSG_RCVD
528408.584 CMAC_LOG_RNG_REQ_TRANSMITTED
528408.586 CMAC_LOG_RNG_RSP_MSG_RCVD
528414.102 CMAC_LOG_WATCHDOG_TIMER
528418.654 CMAC_LOG_RNG_REQ_TRANSMITTED
528418.656 CMAC_LOG_RNG_RSP_MSG_RCVD
528428.726 CMAC_LOG_RNG_REQ_TRANSMITTED
528428.728 CMAC_LOG_RNG_RSP_MSG_RCVD
528438.796 CMAC_LOG_RNG_REQ_TRANSMITTED
528438.798 CMAC_LOG_RNG_RSP_MSG_RCVD
528444.102 CMAC_LOG_WATCHDOG_TIMER
528444.492 CMAC_LOG_LINK_DOWN
528444.494 CMAC_LOG_RESET_FROM_DRIVER
528444.494 CMAC_LOG_STATE_CHANGE                                wait_for_link_up_state
528444.494 CMAC_LOG_DRIVER_INIT_IDB_SHUTDOWN                0x08098D02
528444.494 CMAC_LOG_LINK_DOWN
528474.494 CMAC_LOG_WATCHDOG_TIMER

```

```
528504.494 CMAC_LOG_WATCHDOG_TIMER
528534.494 CMAC_LOG_WATCHDOG_TIMER
```

0 events dropped due to lack of a chunk

The line “0 events dropped due to lack of a chunk” at the end of a display indicates that no log entries were discarded due to a temporary lack of memory, which means the log is accurate and reliable.

The following example compares the output of the **debug cable-modem mac log** command with the **debug cable-modem mac log verbose** command. The **verbose** keyword displays periodic events such as ranging.

```
Router# debug cable-modem mac log

Cable Modem mac log debugging is on
Router#

Router# debug cable-modem mac log verbose

Cable Modem mac log debugging is on (verbose)
Router#
574623.810 CMAC_LOG_RNG_REQ_TRANSMITTED
574623.812 CMAC_LOG_RNG_RSP_MSG_RCVD
574627.942 CMAC_LOG_WATCHDOG_TIMER
574633.880 CMAC_LOG_RNG_REQ_TRANSMITTED
574633.884 CMAC_LOG_RNG_RSP_MSG_RCVD
574643.950 CMAC_LOG_RNG_REQ_TRANSMITTED
574643.954 CMAC_LOG_RNG_RSP_MSG_RCVD
574654.022 CMAC_LOG_RNG_REQ_TRANSMITTED
574654.024 CMAC_LOG_RNG_RSP_MSG_RCVD
574657.978 CMAC_LOG_WATCHDOG_TIMER
574664.094 CMAC_LOG_RNG_REQ_TRANSMITTED
574664.096 CMAC_LOG_RNG_RSP_MSG_RCVD
574674.164 CMAC_LOG_RNG_REQ_TRANSMITTED
574674.166 CMAC_LOG_RNG_RSP_MSG_RCVD

Router# no debug cable-modem mac log verbose

Cable Modem mac log debugging is off
Router#
574684.234 CMAC_LOG_RNG_REQ_TRANSMITTED
574684.238 CMAC_LOG_RNG_RSP_MSG_RCVD
```

The following is sample output from the **debug cable-modem mac messages** command. This command causes received cable MAC management messages to be displayed in a verbose format.

```
Router# debug cable-modem mac messages

dynsrv  dynamic service mac messages
map     map messages received
reg-req reg-req messages transmitted
reg-rsp reg-rsp messages received
rng-req rng-req messages transmitted
rng-rsp rng-rsp messages received
sync    Sync messages received
ucc-req ucc-req messages received
ucc-rsp ucc-rsp messages transmitted
ucd     UCD messages received
<cr>
```

The **dynsrv** keyword displays Dynamic Service Add or Dynamic Service Delete messages during the off-hook/on-hook transitions of a phone connected to the Cisco uBR900 series cable access router.

In addition, sent REG-REQ messages are displayed in hexadecimal dump format. The output from this command is very verbose and is usually not needed for normal interface debugging. The command is most useful when attempting to attach a Cisco uBR900 series cable access router to a cable modem termination system (CMTS) that is not Data-over-Cable Service Interface Specifications (DOCSIS)-qualified.

For a description of the displayed fields of each message, refer to the DOCSIS Radio Frequency Interface Specification, v1.0 (SP-RFI-I04-980724).

Router# **debug cable mac messages**

```
*Mar 7 01:44:06:
*Mar 7 01:44:06: UCD MESSAGE
*Mar 7 01:44:06: -----
*Mar 7 01:44:06:   FRAME HEADER
*Mar 7 01:44:06:     FC                               - 0xC2 == MAC Management
*Mar 7 01:44:06:     MAC_PARM                         - 0x00
*Mar 7 01:44:06:     LEN                               - 0xD3
*Mar 7 01:44:06:   MAC MANAGEMENT MESSAGE HEADER
*Mar 7 01:44:06:     DA                               - 01E0.2F00.0001
*Mar 7 01:44:06:     SA                               - 00E0.1EA5.BB60
*Mar 7 01:44:06:     msg LEN                          - C1
*Mar 7 01:44:06:     DSAP                             - 0
*Mar 7 01:44:06:     SSAP                             - 0
*Mar 7 01:44:06:     control                          - 03
*Mar 7 01:44:06:     version                          - 01
*Mar 7 01:44:06:     type                             - 02 == UCD
*Mar 7 01:44:06:     RSVD                             - 0
*Mar 7 01:44:06:     US Channel ID                    - 1
*Mar 7 01:44:06:     Configuration Change Count      - 4
*Mar 7 01:44:06:     Mini-Slot Size                   - 8
*Mar 7 01:44:06:     DS Channel ID                    - 1
*Mar 7 01:44:06:     Symbol Rate                      - 8
*Mar 7 01:44:06:     Frequency                        - 20000000
*Mar 7 01:44:06:     Preamble Pattern                 - CC CC CC CC CC CC CC CC CC CC CC CC
CC 0D 0D
*Mar 7 01:44:06:   Burst Descriptor 0
*Mar 7 01:44:06:     Interval Usage Code              - 1
*Mar 7 01:44:06:     Modulation Type                  - 1 == QPSK
*Mar 7 01:44:06:     Differential Encoding              - 2 == OFF
*Mar 7 01:44:06:     Preamble Length                   - 64
*Mar 7 01:44:06:     Preamble Value Offset             - 56
*Mar 7 01:44:06:     FEC Error Correction               - 0
*Mar 7 01:44:06:     FEC Codeword Info Bytes           - 16
*Mar 7 01:44:06:     Scrambler Seed                    - 0x0152
*Mar 7 01:44:06:     Maximum Burst Size                 - 1
*Mar 7 01:44:06:     Guard Time Size                   - 8
*Mar 7 01:44:06:     Last Codeword Length              - 1 == FIXED
*Mar 7 01:44:06:     Scrambler on/off                  - 1 == ON
*Mar 7 01:44:06:   Burst Descriptor 1
*Mar 7 01:44:06:     Interval Usage Code              - 3
*Mar 7 01:44:06:     Modulation Type                  - 1 == QPSK
*Mar 7 01:44:06:     Differential Encoding              - 2 == OFF
*Mar 7 01:44:06:     Preamble Length                   - 128
*Mar 7 01:44:06:     Preamble Value Offset             - 0
*Mar 7 01:44:06:     FEC Error Correction               - 5
*Mar 7 01:44:06:     FEC Codeword Info Bytes           - 34
*Mar 7 01:44:06:     Scrambler Seed                    - 0x0152
*Mar 7 01:44:06:     Maximum Burst Size                 - 0
*Mar 7 01:44:06:     Guard Time Size                   - 48
```

```

*Mar 7 01:44:06:      Last Codeword Length      - 1 == FIXED
*Mar 7 01:44:06:      Scrambler on/off          - 1 == ON
*Mar 7 01:44:06:      Burst Descriptor 2
*Mar 7 01:44:06:      Interval Usage Code        - 4
*Mar 7 01:44:06:      Modulation Type            - 1 == QPSK
*Mar 7 01:44:06:      Differential Encoding      - 2 == OFF
*Mar 7 01:44:06:      Preamble Length           - 128
*Mar 7 01:44:06:      Preamble Value Offset     - 0
*Mar 7 01:44:06:      FEC Error Correction       - 5
*Mar 7 01:44:06:      FEC Codeword Info Bytes    - 34
*Mar 7 01:44:06:      Scrambler Seed            - 0x0152
*Mar 7 01:44:06:      Maximum Burst Size        - 0
*Mar 7 01:44:06:      Guard Time Size           - 48
*Mar 7 01:44:06:      Last Codeword Length      - 1 == FIXED
*Mar 7 01:44:06:      Scrambler on/off          - 1 == ON
*Mar 7 01:44:06:      Burst Descriptor 3
*Mar 7 01:44:06:      Interval Usage Code        - 5
*Mar 7 01:44:06:      Modulation Type            - 1 == QPSK
*Mar 7 01:44:06:      Differential Encoding      - 2 == OFF
*Mar 7 01:44:06:      Preamble Length           - 72
*Mar 7 01:44:06:      Preamble Value Offset     - 48
*Mar 7 01:44:06:      FEC Error Correction       - 5
*Mar 7 01:44:06:      FEC Codeword Info Bytes    - 75
*Mar 7 01:44:06:      Scrambler Seed            - 0x0152
*Mar 7 01:44:06:      Maximum Burst Size        - 0
*Mar 7 01:44:06:      Guard Time Size           - 8
*Mar 7 01:44:06:      Last Codeword Length      - 1 == FIXED
*Mar 7 01:44:06:      Scrambler on/off          - 1 == ON
*Mar 7 01:44:06:
*Mar 7 01:44:06:      MAP MESSAGE
*Mar 7 01:44:06:      -----
*Mar 7 01:44:06:      FRAME HEADER
*Mar 7 01:44:06:      FC                          - 0xC3 == MAC Management with Extended
Header
*Mar 7 01:44:06:      MAC_PARM                    - 0x02
*Mar 7 01:44:06:      LEN                          - 0x42
*Mar 7 01:44:06:      EHDR                        - 0x00 0x00
*Mar 7 01:44:06:      MAC MANAGEMENT MESSAGE HEADER
*Mar 7 01:44:06:      DA                          - 01E0.2F00.0001
.
*Mar 7 01:44:17:      RNG-RSP MESSAGE
*Mar 7 01:44:17:      -----
*Mar 7 01:44:17:      FRAME HEADER
*Mar 7 01:44:17:      FC                          - 0xC2 == MAC Management
*Mar 7 01:44:17:      MAC_PARM                    - 0x00
*Mar 7 01:44:17:      LEN                          - 0x2B
*Mar 7 01:44:17:      MAC MANAGEMENT MESSAGE HEADER
*Mar 7 01:44:17:      DA                          - 00F0.1EB2.BB61
.
*Mar 7 01:44:20:      REG-REQ MESSAGE
*Mar 7 01:44:20:      -----
*Mar 7 01:44:20:      C20000A5 000000E0 1EA5BB60 00F01EB2
*Mar 7 01:44:20:      BB610093 00000301 06000004 03010104
*Mar 7 01:44:20:      1F010101 0204003D 09000304 001E8480
*Mar 7 01:44:20:      04010705 04000186 A0060200 0C070101
*Mar 7 01:44:20:      080300F0 1E112A01 04000000 0A020400
*Mar 7 01:44:20:      00000A03 04000002 58040400 00000105
*Mar 7 01:44:20:      04000000 01060400 00025807 04000000
*Mar 7 01:44:20:      3C2B0563 6973636F 06105E4F C908C655
*Mar 7 01:44:20:      61086FD5 5C9D756F 7B730710 434D5453
*Mar 7 01:44:20:      204D4943 202D2D2D 2D2D2D2D 0C040000
*Mar 7 01:44:20:      00000503 010100
*Mar 7 01:44:20:

```

```

*Mar 7 01:44:20:
*Mar 7 01:44:20: REG-RSP MESSAGE
*Mar 7 01:44:20: -----
*Mar 7 01:44:20:   FRAME HEADER
*Mar 7 01:44:20:     FC                               - 0xC2 == MAC Management
*Mar 7 01:44:20:     MAC_PARM                          - 0x00
*Mar 7 01:44:20:     LEN                                - 0x29
*Mar 7 01:44:20:   MAC MANAGEMENT MESSAGE HEADER
*Mar 7 01:44:20:     DA                                - 00F0.1EB2.BB61

```

---

**Related Commands**

Command	Description
<b>debug cable-modem bpkm</b>	Debugs baseline privacy information.
<b>debug cable-modem bridge</b>	Displays bridge filter processing information.
<b>debug cable-modem error</b>	Enables debugging messages for the cable interface driver.
<b>debug cable-modem interrupts</b>	Debugs cable modem interrupts.
<b>debug cable-modem map</b>	Displays the timing from MAP messages to synchronize messages and the timing between MAP messages on a Cisco uBR900 series cable access router.

---

# debug cable-modem map

To display the timing from MAP messages to synchronized messages and the timing between MAP messages on a Cisco uBR900 series cable access router, use the **debug cable-modem map** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable-modem map**

**no debug cable-modem map**

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

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3 NA	This command was introduced.

**Examples** The following is sample output from the **debug cable-modem map** command:

```
Router# debug cable-modem map

Cable Modem MAP debugging is on
Router#
*Mar 7 20:12:08: 595322.942: Min MAP to sync=72
*Mar 7 20:12:08: 595322.944: Max map to map time is 40
*Mar 7 20:12:08: 595322.982: Min MAP to sync=63
*Mar 7 20:12:08: 595323.110: Max map to map time is 41
*Mar 7 20:12:08: 595323.262: Min MAP to sync=59
*Mar 7 20:12:08: 595323.440: Max map to map time is 46
*Mar 7 20:12:09: 595323.872: Min MAP to sync=58
```

Related Commands	Command	Description
	<a href="#">debug cable-modem bpkm</a>	Debugs baseline privacy information.
	<a href="#">debug cable-modem bridge</a>	Displays bridge filter processing information.
	<a href="#">debug cable-modem error</a>	Enables debugging messages for the cable interface driver.
	<a href="#">debug cable-modem interrupts</a>	Debugs cable modem interrupts.
	<a href="#">debug cable-modem mac</a>	Troubleshoots the MAC layer for cable modems.

# debug cable phy

To activate debugging of messages generated in the cable physical layer, use the **debug cable phy** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable phy**

**no debug cable phy**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command activates debugging of messages generated in the cable physical sublayer (PHY), which is where upstream and downstream activity between the Cisco uBR7246 router and the hybrid fiber-coaxial (HFC) network is controlled. When this command is activated, any messages generated in the cable PHY will be displayed on the Cisco uBR7246 console.

---

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

```
Router# debug cable phy

cmts_phy_init: mac_version == BCM3210_FPGA
bcm3033_set_tx_sym_rate(5056941)
stintctl = 0x54484800
bcm3033_set_tx_if_freq(44000000)
stfreqctl = 0x5BAAAAAA
cmts_phy_init_us: U0 part_id = 0x3136, rev_id = 0x05, rev_id2 = 0x64
cmts_phy_init: mac_version == BCM3210_FPGA
Media access controller chip version.
bcm3033_set_tx_sym_rate(5056941)
stintctl = 0x54484800
Physical layer symbol rate register value.
00:51:49: bcm3033_set_tx_if_freq(44000000)
00:51:49: stfreqctl = 0x5BAAAAAA
Physical layer intermediate frequency (IF) register value.
00:51:49: cmts_phy_init_us: U0 part_id = 0x3136, rev_id = 0x05, rev_id2 = 0x64
Physical layer receiver chip part version.
```

# debug cable privacy

To activate debugging of baseline privacy, use the **debug cable privacy** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable privacy**

**no debug cable privacy**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command activates debugging of baseline privacy. When this command is activated, any messages generated by the spectrum manager will be displayed on the Cisco uBR7246 console.

---

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

```
Router# debug cable privacy

Removing both odd and even keys for sid %x.

    Invalid Len for TLV_SERIAL_NUM_TYPE : %d.

    Invalid Len for TLV_MANUF_ID_TYPE : %d.

    Invalid Len for TLV_MANUF_ID_TYPE : %d.
```

# debug cable qos

To activate quality of service (QoS) debugging, use the **debug cable qos** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable qos**

**no debug cable qos**

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

**Command Modes** Privileged EXEC

**Usage Guidelines** This command activates debugging of QoS. When this command is activated, any messages related to QoS parameters will be displayed on the Cisco uBR7246 console.

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

```
Router# debug cable qos

CMTS_QOS_LOG_NO_MORE_QOS_INDEX
Modems cannot add more entries to the class of service table.
CMTS_QOS_LOG_NOMORE_QOSPRF_MEM
Memory allocation error when creating class of service table entry.
CMTS_QOS_LOG_NO_CREATION_ALLOWED
Class of service entry cannot be created by modem. Use CLI or SNMP
interface instead of the modem's TFTP configuration file.
CMTS_QOS_LOG_CANNOT_REGISTER_COS_SID
A service identifier (SID) could not be assigned to the registering modem.
CMTS_QOS_LOG_CANNOT_DEREGISTER_COS_SID
The modem's service identifier (SID) was already removed.
CMTS_QOS_LOG_MSLOT_TIMEBASE_WRAPPED
The 160 KHz timebase clock drives a 26-bit counter which wraps around
approximately every 7 minutes. This message is generated every time it
wraps around.
```

# debug cable range

To display ranging messages from cable modems on the hybrid fiber-coaxial (HFC) network, use the **debug cable range** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable range**

**no debug cable range**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command activates debugging of ranging messages from cable modems on the hybrid fiber-coaxial (HFC) network. When this command is activated, any ranging messages generated when cable modems request or change their upstream frequencies will be displayed on the Cisco uBR7246 console. Use this command to display the details of the initial and station maintenance procedures. The initial maintenance procedure is used for link establishment. The station maintenance procedure is used for link keepalive monitoring.

---

**Examples** The following is sample output from the **debug cable range** command when a modem first seeks to establish a link to the Cisco uBR7246 universal broadband router:

```
Router# debug cable range

Got a ranging request
SID value is 0 on Interface Cable3/0/U0
CM mac address 00:10:7B:43:AA:21 Timing offset is 3312
3E 1E 3F FF 00 00 59 BF 01 15 F8 01 A7 00 0C F0
```

The SID value of 0 indicates that the modem has no assigned service identifier. The “CM mac address” is the MAC address of the radio frequency (RF) interface of the modem, not its Ethernet interface. The “Timing offset” is a measure of the distance between the modem and the Cisco uBR7246 universal broadband router expressed in 10.24-MHz clocks. This value is adjusted down to zero by the maintenance procedures. The first 16 bytes of the prepended header of the message are dumped in hexadecimal.

The following is sample output when the modem is first assigned a SID during initial maintenance:

```
CM mac address 0010.7b43.aa21
  found..Assigned SID #2 on Interface Cable3/0/U0
  Timing offset is CF0
  Power value is 15F8, or -1 dB
  Freq Error = 423, Freq offset is 1692
  Ranging Modem with Sid 2 on i/f : Cable3/0/U0
```

The following is sample output when the modem is reassigned the same SID during initial maintenance:

```
Initial Range Message Received on Interface Cable3/0/U0
CMTS reusing old sid : 2 for modem : 0010.7b43.aa21
Timing offset is CF0
```

```
Power value is 15F8, or -1 dB  
Freq Error = 423, Freq offset is 1692  
Ranging Modem with Sid 2 on i/f : Cable3/0/U0
```

The following is sample output when the modem is polled by the Cisco uBR7246 universal broadband router during station maintenance. Polling happens at a minimum rate of once every 10 seconds.

```
Ranging Modem with Sid 2 on i/f : Cable3/0/U0
```

```
Got a ranging request  
SID value is 2 on Interface Cable3/0/U0  
CM mac address 00:10:7B:43:AA:21  
Timing offset is 0  
Power value is 1823, or -1 dB  
Freq Error = 13, Freq offset is 0  
Ranging has been successful for SID 2 on Interface Cable3/0/U0
```

# debug cable reset

To display reset messages from cable interfaces, use the **debug cable reset** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable reset**

**no debug cable reset**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command activates display of reset messages from cable interfaces.

---

**Examples** The following is sample output from the **debug cable reset** command when the interface is reset due to complete loss of receive packets:

```
Router# debug cable reset
```

```
Resetting CMTS interface.
```

# debug cable specmgmt

To debug spectrum management (frequency agility) on the hybrid fiber-coaxial (HFC) network, use the **debug cable specmgmt** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable specmgmt**

**no debug cable specmgmt**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command activates debugging of spectrum management (frequency agility) on the HFC network. When this command is activated, any messages generated due to spectrum group activity will be displayed on the Cisco uBR7246 console. Spectrum group activity can be additions or changes to spectrum groups, or frequency and power level changes controlled by spectrum groups.

---

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

```
Router# debug cable specmgmt  
  
cmts_next_frequency(0x60A979AC, 1, 1)
```

The following is sample output when the frequency hop was commanded:

```
add_interface_to_freq(0x60BD3734, 0x60C44F68)
```

The following is sample output when the interface was added to the interface list of a frequency:

```
set_upstream(0x60A979AC, 1, 21000000, -5)
```

The following is sample output when the spectrum management has set the frequency and power level of an upstream port:

```
cmts_frequency_hop_decision(0x60B57FEC)
```

# debug cable startalloc

To debug channel allocations on the hybrid fiber-coaxial (HFC) network, use the **debug cable startalloc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable startalloc**

**no debug cable startalloc**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command activates debugging of any channel allocations on the HFC network. When this command is activated, any messages generated when channels are allocated to cable modems on the HFC network will be displayed on the Cisco uBR7246 console.

---

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

```
Router# debug cable startalloc  
  
MAP startalloc adjusted by <n> mslots
```

This output indicates time-slot Manufacturing Automation Protocol (MAP) processing is active.

# debug cable telco-return

To display debug messages for Telco return events, use the **debug cable telco-return** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable telco-return**

**no debug cable telco-return**

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

**Defaults** Debugging for Telco return events is not enabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(4)XI	This command was introduced.

**Examples** The following is sample output from the **debug cable telco-return** command:

```
Router# debug cable telco-return

CMTS telco-return debugging is on
```

Related Commands	Command	Description
	<a href="#">debug cable ucc</a>	Debugs upstream channel change messages generated when cable modems request or are assigned a new channel.

# debug cable ucc

To debug upstream channel change (UCC) messages generated when cable modems request or are assigned a new channel, use the **debug cable ucc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable ucc**

**no debug cable ucc**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command activates debugging of any UCC messages generated when cable modems request or are assigned a new channel. When this command is activated, any messages related to upstream channel changes will be displayed on the Cisco uBR7246 console.

---

**Examples** The following is sample output from the **debug cable ucc** command when moving a modem from one upstream channel to another:

```
Router# debug cable ucc

SID 2 has been registered

Mac Address of CM for UCC
    00:0E:1D:D8:52:16

UCC Message Sent to CM

Changing SID 2 from upstream channel 1 to upstream channel 2
```

# debug cable ucd

To debug upstream channel descriptor (UCD) messages, use the **debug cable ucd** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cable ucd**

**no debug cable ucd**

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

**Command Modes** Privileged EXEC

**Usage Guidelines** This command activates debugging of any UCD messages. UCD messages contain information about upstream channel characteristics and are sent to the cable modems on the hybrid fiber-coaxial (HFC) network. Cable modems that are configured to use enhanced upstream channels use these UCD messages to identify and select an enhanced upstream channel to use. When this command is activated, any messages related to upstream channel descriptors will be displayed on the Cisco uBR7246 console.

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

```
Router# debug cable ucd

UCD MESSAGE
-----
FRAME HEADER
FC - 0xC2 ==
MAC_PARM - 0x00
LEN - 0xD3
MAC MANAGEMENT MESSAGE HEADER
DA - 01E0.2F00.0001
SA - 0009.0CEF.3730
msg LEN - C1
DSAP - 0
SSAP t - 0
control - 03
version - 01
type - 02 ==
US Channel ID - 1
Configuration Change Count - 5
Mini-Slot Size - 4
DS Channel ID - 1
Symbol Rate - 8
Frequency - 10000000
Preamble Pattern - CC CC CC CC CC CC CC CC CC CC CC
CC 0D 0D
Burst Descriptor 0
Interval Usage Code - 1
Modulation Type - 1 == QPSK
Differential Encoding - 2 == OFF
Preamble Length - 64
Preamble Value Offset - 56
FEC Error Correction - 0
```

```

FEC Codeword Length      - 16
Scrambler Seed           - 0x0152
Maximum Burst Size      - 2
Guard Time Size         - 8
Last Codeword Length    - 1 == FIXED
Scrambler on/off        - 1 == ON
Burst Descriptor 1
  Interval Usage Code    - 3
  Modulation Type        - 1 == QPSK
  Differential Encoding   - 2 == OFF
  Preamble Length        - 128
  Preamble Value Offset  - 0
  FEC Error Correction    - 5
  FEC Codeword Length    - 34
  Scrambler Seed         - 0x0152
  Maximum Burst Size     - 0
  Guard Time Size        - 48
  Last Codeword Length   - 1 == FIXED
  Scrambler on/off       - 1 == ON
Burst Descriptor 2
  Interval Usage Code    - 4
  Modulation Type        - 1 == QPSK
  Differential Encoding   - 2 == OFF
  Preamble Length        - 128
  Preamble Value Offset  - 0
  FEC Error Correction    - 5
  FEC Codeword Length    - 34
  Scrambler Seed         - 0x0152
  Maximum Burst Size     - 0
  Guard Time Size        - 48
  Last Codeword Length   - 1 == FIXED
  Scrambler on/off       - 1 == ON
Burst Descriptor 3
  Interval Usage Code    - 5
  Modulation Type        - 1 == QPSK
  Differential Encoding   - 2 == OFF
  Preamble Length        - 72
  Preamble Value Offset  - 48
  FEC Error Correction    - 5
  FEC Codeword Length    - 75
  Scrambler Seed         - 0x0152
  Maximum Burst Size     - 0
  Guard Time Size        - 8
  Last Codeword Length   - 1 == FIXED
  Scrambler on/off       - 1 == ON

```

The UCD MESSAGE is :

```

0xC2 0x00 0x00 0xD3 0x00 0x00 0x01 0xE0
0x2F 0x00 0x00 0x01 0x00 0x09 0x0C 0xEF
0x37 0x30 0x00 0xC1 0x00 0x00 0x03 0x01
0x02 0x00 0x01 0x05 0x04 0x01 0x01 0x01
0x08 0x02 0x04 0x00 0x98 0x96 0x80 0x03
0x10 0xCC 0xCC 0xCC 0xCC 0xCC 0xCC 0xCC
0xCC 0xCC 0xCC 0xCC 0xCC 0xCC 0xCC 0x0D
0x0D 0x04 0x25 0x01 0x01 0x01 0x01 0x02
0x01 0x02 0x03 0x02 0x00 0x40 0x04 0x02
0x00 0x38 0x05 0x01 0x00 0x06 0x01 0x10
0x07 0x02 0x01 0x52 0x08 0x01 0x02 0x09
0x01 0x08 0x0A 0x01 0x01 0x0B 0x01 0x01
0x04 0x25 0x03 0x01 0x01 0x01 0x02 0x01
0x02 0x03 0x02 0x00 0x80 0x04 0x02 0x00
0x00 0x05 0x01 0x05 0x06 0x01 0x22 0x07
0x02 0x01 0x52 0x08 0x01 0x00 0x09 0x01
0x30 0x0A 0x01 0x01 0x0B 0x01 0x01 0x04

```

```
0x25 0x04 0x01 0x01 0x01 0x02 0x01 0x02
0x03 0x02 0x00 0x80 0x04 0x02 0x00 0x00
0x05 0x01 0x05 0x06 0x01 0x22 0x07 0x02
0x01 0x52 0x08 0x01 0x00 0x09 0x01 0x30
0x0A 0x01 0x01 0x0B 0x01 0x01 0x04 0x25
0x05 0x01 0x01 0x01 0x02 0x01 0x02 0x03
0x02 0x00 0x48 0x04 0x02 0x00 0x30 0x05
0x01 0x05 0x06 0x01 0x4B 0x07 0x02 0x01
0x52 0x08 0x01 0x00 0x09 0x01 0x08 0x0A
0x01 0x01 0x0B 0x01 0x01
```

# debug callback

To display callback events when the router is using a modem and a chat script to call back on a terminal line, use the **debug callback** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug callback**

**no debug callback**

---

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

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command is useful for debugging chat scripts on PPP and AppleTalk Remote Access Protocol (ARAP) lines that use callback mechanisms. The output provided by the **debug callback** command shows you how the call is progressing when used with the **debug ppp** or **debug arap** commands.

---

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

```
Router# debug callback

TTY7 Callback process initiated, user: exec_test dialstring 123456
TTY7 Callback forced wait = 4 seconds
TTY7 Exec Callback Successful - await exec/autoselect pickup
TTY7: Callback in effect
```

---

Related Commands	Command	Description
	<a href="#">debug cable env</a>	Displays ARAP events.
	<a href="#">debug ppp</a>	Displays information on traffic and exchanges in an internetwork implementing the PPP.

---

# debug call fallback detail

To display details of the call fallback, use the **debug call fallback detail** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call fallback detail**

**no debug call fallback detail**

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

**Defaults** Disabled

**Command Modes** Privileged EXEC

Release	Modification
12.1(3)T	This command was introduced.
12.2(2)XB1	This command was implemented on the Cisco AS5850 platform.
12.2(4)T	This command was implemented on the Cisco 7200 series routers.
12.2(4)T3	This command was implemented on the Cisco 7500 series routers routers.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T.

**Usage Guidelines** Every time a call request is received, the **debug call fallback detail** command displays in the command-line interface (CLI) cache lookup and call acceptance/rejection information. Use this command to monitor call requests as they enter the call fallback subsystem.

If you have a large amount of calls in your router, enabling this command can cause delays in your routing functions as the debug statistics are constantly compiled and sent to your terminal. Also, debug messages on your terminal may make for difficult CLI configuring.

**Examples** The following example depicts a call coming in to 10.1.1.4 with codec g729r8. Because there is no cache entry for this destination, a probe is sent and values are inserted into the cache. A lookup is performed again, entry is found, and a fallback decision is made to admit the call.

```
Router# debug call fallback detail
```

```
When cache is empty:  
debug call fallback detail:  
2d19h:fb_lookup_cache:10.1.1.4, codec:g729r8  
2d19h:fb_lookup_cache:No entry found.  
2d19h:fb_check:no entry exists, enqueueing probe info... 10.1.1.4, codec:g729r8  
2d19h:fb_main:Got FB_APP_INQ event  
2d19h:fb_main:Dequeued prob info: 10.1.1.4, codec:g729r8  
2d19h:fb_lookup_cache:10.1.1.4, codec:g729r8  
2d19h:fb_lookup_cache:No entry found.  
2d19h:fb_cache_insert:insert:10.1.1.4, codec:g729r8  
2d19h:fb_cache_insert:returning entry:10.1.1.4, codec:g729r8  
2d19h:fb_initiate_probe:Creating probe... 10.1.1.4, codec:g729r8  
2d19h:fb_initiate_probe:Created and started on probe #13, 10.1.1.4, codec:g729r8  
2d19h:fb_lookup_cache:10.1.1.4, codec:g729r8  
2d19h:fb_lookup_cache:Found entry.  
2d19h:fb_check:returned FB_CHECK_TRUE, 10.1.1.4, codec:g729r8  
2d19h:fb_main:calling callback function with:TRUE
```

The following example depicts a call coming in to 10.1.1.4 with codec g729r8. A lookup is performed, entry is found, and a fallback decision is made to admit the call.

```
Router# debug call fallback detail
```

```
When cache is full:  
2d19h:fb_lookup_cache:10.1.1.4, codec:g729r8  
2d19h:fb_lookup_cache:Found entry.  
2d19h:fb_check:returned FB_CHECK_TRUE, 10.1.1.4, codec:g729r8  
2d19h:fb_main:calling callback function with:TRUE
```

# debug call fallback probe

To display details of the call fallback probes, use the **debug call fallback probe** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call fallback probe**

**no debug call fallback probe**

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

**Defaults** Disabled

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(3)T	This command was introduced.
	12.2(2)XA	The <b>call fallback</b> and <b>call fallback reject-cause-code</b> commands were introduced.
	12.2(2)XB1	This command was implemented on the Cisco AS5850 platform.
	12.2(4)T	This command was implemented on the Cisco 7200 series routers.
	12.2(4)T3	This command was implemented on the Cisco 7500 series routers.

**Usage Guidelines** Every time a probe is received, the **debug call fallback probe** command displays in the command-line interface (CLI) network traffic information collected by the probe. Use this command to monitor the network traffic information the probes carry as they enter the call fallback subsystem and log cache entries.

If you have frequent return of probes to your router, enabling this command can cause delays in your routing functions as the debug statistics are constantly compiled and sent to your terminal. Also, debug messages on your terminal may make for difficult CLI configuring.

**Examples** The following example depicts a call coming in to 10.1.1.4 and codec type g729r8. Because there is no cache entry for this IP address, a g729r8 probe is initiated. The probe consists of 20 packet returns with an average delay of 43 milliseconds. The “jitter out” is jitter from source to destination router and “jitter in” is jitter from destination to source router. The delay, loss, and Calculated Planning Impairment Factor (ICPIF) values following g113\_calc\_icpif are the instantaneous values, whereas those values following “New smoothed values” are the values after applying the smoothing with weight 65.

```
Router# debug call fallback probe

2d19h:fb_initiate_probe:Probe payload is 32
2d19h:fb_main:NumOfRTT=20, RTTSum=120, loss=0, delay=43, jitter in=0, jitter out=0->
10.1.1.4, codec:g729r8
2d19h:g113_calc_icpif(delay (w/codec delay)=43, loss=0, expect_factor=10) Icpif=0
```

```
2d19h:fb_main:Probe timer expired, 10.1.1.4, codec:g729r8
2d19h:fb_main:NumOfRTT=20, RTTSum=120, loss=0, delay=43, jitter in=0, jitter out=0->
10.1.1.4, codec:g729r8
2d19h:g113_calc_icpif(delay (w/codec delay)=43, loss=0, expect_factor=10) Icpif=0
2d19h:fb_main:New smoothed values:inst_weight=65, ICPIF=0, Delay=43, Loss=0 -> 10.1.1.4,
codec:g729r8
```

# debug call filter detail

To display details of the debug trace inside the generic call filter module (GCFM), use the **debug call filter detail** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call filter detail**

**no debug call filter detail**

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

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.3(4)T	This command was introduced.

**Examples** The following sample output from the **debug call filter detail** command shows the detailed activity of the GCFM, which is the internal module that controls the debug filtering.

```
Router# debug call filter detail

5d18h: gcfm_call_get_hash_address: hashtable index = 345
5d18h: gcfm_call_search_hash:no found
5d18h: gcfm_init_call_record:
5d18h: gcfm_init_percall_matchlist:
5d18h: === list 1: service_state=2, callp's: 0
5d18h: gcfm_call_get_hash_address: hashtable index = 345
5d18h: gcfm_call_enlist: count before this enlist 0 on 624D6000
5d18h: gcfm_call_enlist: tail is empty guid=C2E4C789-214A-11D4-804C-000A8A389BA8
5d18h: gcfm_call_get_hash_address: hashtable index = 345
5d18h: gcfm_call_search_hash: search requested guid=C2E4C789-214A-11D4-804C-000A8A389BA8
vs the entry guid=C2E4C789-214A-11D4-804C-000A8A389BA8
5d18h: gcfm_call_search_hash: found
5d18h: gcfm_update_percall_condlist_context:
5d18h: gcfm_update_percall_condlist_context: check cond = 2
5d18h: gcfm_copy_match_cond:
5d18h: gcfm_update_cond_through_matchlist:
5d18h: gcfm_check_percond_with_matchlist: check match-list 1
5d18h: gcfm_matchlist_percond_check:
5d18h: gcfm_matchlist_percond_check: check cond=2
5d18h: gcfm_matchlist_percond_check: compare 42300 to configured 42300
5d18h: gcfm_check_cond_tel_number:
5d18h: gcfm_check_cond_tel_number: matched
5d18h: gcfm_matchlist_percond_check: checked result is 1
5d18h: gcfm_is_bitfield_identical:
```

```
5d18h: gcfm_update_cond_through_matchlist: service=1, percallmatchlist
tag=1,current_status = 1, service_filter=0
5d18h: gcfm_perall_notify_condition: not linked call record
```

Table 33 describes the significant fields shown in the display.

**Table 33** *debug call filter detail Field Descriptions*

Field	Description
5d18h: gcfm_init_perall_matchlist:	Shows that the filtering has been initiated.
5d18h: gcfm_call_enlist: tail is empty guid=C2E4C789-214A-11D4-804C-00 0A8A389BA8	Shows the global unique identifier (GUID) for the call.
5d18h: gcfm_check_percond_with_matchlist: check match-list 1	Shows which match list is being checked.
5d18h: gcfm_matchlist_percond_check: checked result is 1	Shows that the call matched conditions in match list 1.

**Related Commands**

Command	Description
<b>debug call filter inout</b>	Displays the debug trace inside the GCFM.
<b>debug condition match-list</b>	Runs a filtered debug on a voice call.
<b>show call filter components</b>	Displays the components used for filtering calls.

# debug call filter inout

To display the debug trace inside the generic call filter module (GCFM), use the **debug call filter inout** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call filter inout**

**no debug call filter inout**

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

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.3(4)T	This command was introduced.

**Examples** The following sample output from the **debug call filter inout** command shows the incoming and outgoing activity of the GCFM, which is the internal module that controls the debug filtering.

```
Router# debug call filter inout

5d18h: gcfm_generate_guid: component ISDN gets guid
5d18h: gcfm_percall_register: component ISDN
5d18h: gcfm_percall_register: component ISDN return selected=0
5d18h: gcfm_percall_notify_condition: component ISDN for sync=1
5d18h: gcfm_percall_notify_condition: component ISDN successfully selected = 0
5d18h: gcfm_check_percall_status: component TGRM
5d18h: gcfm_check_percall_status: component TGRM return selected=0
5d18h: gcfm_check_percall_status: component TGRM
5d18h: gcfm_check_percall_status: component TGRM return selected=0
5d18h: gcfm_percall_register: component VTSP
5d18h: gcfm_percall_register: component VTSP for return selected value 0
5d18h: gcfm_percall_notify_condition: component VTSP for sync=1
5d18h: gcfm_percall_notify_condition: component VTSP successfully selected = 0
5d18h: gcfm_percall_register: component CCAPI
5d18h: gcfm_percall_register: component CCAPI for return selected value 0
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION return selected=0
5d18h: gcfm_percall_register: component VOICE-IVR-V2
5d18h: gcfm_percall_register: component VOICE-IVR-V2 for return selected value 0
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component DIAL-PEER
5d18h: gcfm_check_percall_status: component DIAL-PEER return selected=0
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component DIAL-PEER
5d18h: gcfm_check_percall_status: component DIAL-PEER return selected=0
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
```

```

5d18h: gcfm_check_percall_status: component DIAL-PEER
5d18h: gcfm_check_percall_status: component DIAL-PEER return selected=0
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component DIAL-PEER
5d18h: gcfm_check_percall_status: component DIAL-PEER return selected=0
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component DIAL-PEER
5d18h: gcfm_check_percall_status: component DIAL-PEER return selected=0
5d18h: gcfm_check_percall_status: component DIAL-PEER
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_percall_register: component CCAPI
5d18h: gcfm_percall_register: component CCAPI for return selected value 0
5d18h: gcfm_percall_register: component VOICE-IVR-V2
5d18h: gcfm_percall_register: component VOICE-IVR-V2 for return selected value 0
5d18h: gcfm_percall_notify_condition: component VOICE-IVR-V2 for sync=1
5d18h: gcfm_percall_notify_condition: component VOICE-
Router#IVR-V2 successfully selected = 1
5d18h: gcfm_percall_register: component H323
5d18h: gcfm_percall_register: component H323 for return selected value 1
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION return selected=1
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION return selected=1
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION return selected=1
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION return selected=1
5d18h: gcfm_clear_condition: component VOICE-IVR-V2
5d18h: gcfm_clear_condition: component VOICE-IVR-V2 successfully
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION return selected=0
5d18h: gcfm_percall_deregister: component CCAPI
5d18h: gcfm_percall_deregister: component CCAPI successfully
5d18h: gcfm_percall_deregister: component H323
5d18h: gcfm_percall_deregister: component H323 successfully
5d18h: gcfm_percall_deregister: component ISDN
5d18h: gcfm_percall_deregister: component ISDN successfully
5d18h: gcfm_percall_deregister: component VOICE-IVR-V2
5d18h: gcfm_percall_deregister: component VOICE-IVR-V2 successfully
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION
5d18h: gcfm_check_percall_status: component NUMBER-TRANSLATION return selected=0
5d18h: gcfm_percall_deregister: component CCAPI
5d18h: gcfm_percall_deregister: component CCAPI successfully
5d18h: gcfm_percall_deregister: component VTSP
5d18h: gcfm_percall_deregister: component VTSP successfully
5d18h: gcfm_percall_deregister: component VOICE-IVR-V2
5d18h: gcfm_terminate_track_guid: component VOICE-IVR-V2 terminate, success
5d18h: gcfm_percall_deregister: component VOICE-IVR-V2 successfully

```

Table 34 describes the significant fields shown in the display.

**Table 34** debug call filter inout Field Descriptions

Field	Description
gcfm_generate_guid:	Shows that a GUID has been generated.
gcfm_percall_register:	Shows components that have been registered for the call.
gcfm_percall_notify_condition:	Shows that a component has been notified of the call.
gcfm_check_percall_status:	Shows the status of a component of the call.
gcfm_percall_register:	Shows that a component has been registered.

**Table 34** *debug call filter inout Field Descriptions (continued)*

Field	Description
gcfm_clear_condition:	Shows that a condition is cleared for a component.
gcfm_percall_deregister:	Shows that a component has been deregistered.
gcfm_terminate_track_guid:	Shows that the router is no longer tracking the GUID.

**Related Commands**

Command	Description
<b>debug call filter detail</b>	Displays the details of the debug trace inside the GCFM.
<b>debug condition match-list</b>	Runs a filtered debug on a voice call.
<b>show call filter components</b>	Displays the components used for filtering calls.

# debug call-mgmt

To display debugging information for call accounting, including modem and time slot usage, for active and recent calls, use the **debug call-mgmt** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call-mgmt**

**no debug call-mgmt**

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

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1	This command was introduced.

**Examples** The following is sample output after the **debug call-mgmt** command has been enabled:

```
Router# debug call-mgmt

Call Management debugging is on
Router#
Dec 26 13:57:27.710: msg_to_calls_mgmt: msg type CPM_NEW_CALL_CSM_CONNECT received
Dec 26 13:57:27.714: In actv_c_proc_message,
    access type CPM_INSERT_NEW_CALL,
    call type CPM_ISDN_ANALOG:
        CSM completed connecting a new modem call
.
.
.
Dec 26 13:57:45.906: msg_to_calls_mgmt: msg type CPM_NEW_CALL_ISDN_CONNECT received
Dec 26 13:57:45.906: In actv_c_proc_message,
    access type CPM_INSERT_NEW_CALL,
    call type CPM_ISDN_ANALOG:
        Added a new ISDN analog call to the active-calls list
        CC-Slot#7, DSX1-Ctrlr#17, DS0-Timeslot#1
        Mdm-Slot#1, Mdm-Port#3, TTY#219
.
.
.
Dec 26 13:58:25.682: Call mgmt per minute statistics:
    active list length: 1
    history list length: 3
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 1
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 2
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 3
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 4
```

```

Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 5
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 6
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 7
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 8
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 9
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 10
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 11
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 12
Dec 26 13:58:25.682:      0 timeslots active at slot 7, ctrlr 13
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 14
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 15
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 16
Dec 26 13:58:25.686:      1 timeslots active at slot 7, ctrlr 17
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 18
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 19
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 20
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 21
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 22
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 23
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 24
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 25
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 26
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 27
Dec 26 13:58:25.686:      0 timeslots active at slot 7, ctrlr 28

```

```
Router# clear int as1/03
```

```

Dec 26 13:58:26.538: msg_to_calls_mgmt: msg type CPM_VOICE_CALL_REJ_NO_MOD_AVAIL received
Dec 26 13:58:26.538: In actv_c_proc_message,
    access type CPM_REMOVE_DISC_CALL,
    call type CPM_ISDN_ANALOG:
        Removed a disconnected ISDN analog call
        CC-Slot#7, DSX1-Ctrlr#17, DS0-Timeslot#1
Dec 26 13:58:26.538:      Mdm-Slot#1, Mdm-Port#3, TTY#219

```

Table 35 describes the significant fields shown in the display.

**Table 35** *debug call-mgmt Field Descriptions*

Field	Description
CPM_NEW_CALL_CSM_CONNECT	Indicates the arrival of a new call.
access type CPM_INSERT_NEW_CALL, call type CPM_ISDN_ANALOG:	Indicates that the new call is an analog ISDN B channel call (either a voice call or a call over an analog modem), rather than a digital (V.110) call.
CC-Slot#7, DSX1-Ctrlr#17, DS0-Timeslot#1 Mdm-Slot#1, Mdm-Port#3, TTY#219	Indicates that the call is connected via the B channel on Serial7/17:1 to the asynchronous modem resource 1/03 (interface async1/03, also known as line tty219).
Dec 26 13:58:25.682: Call mgmt per minute statistics: active list length: 1 history list length: 3	Displays periodic statistics that give the allocation state of each DSX1 interface present in the system, as well as the number of current (active) and recent (history) calls.
Dec 26 13:58:26.538: msg_to_calls_mgmt: msg type CPM_VOICE_CALL_REJ_NO_MOD_ AVAIL received	Indicates that the analog ISDN B channel call has been disassociated from a modem.

**Table 35** *debug call-mgmt Field Descriptions (continued)*

Field	Description
access type CPM_REMOVE_DISC_CALL, call type CPM_ISDN_ANALOG: Removed a disconnected ISDN analog call	Indicates that the analog ISDN B channel call has been disconnected.
CC-Slot#7, DSX1-Ctrlr#17, DS0-Timeslot#1 Dec 26 13:58:26.538: Mdm-Slot#1, Mdm-Port#3, TTY#219	Indicates that the call has been disconnected via the B channel on Serial7/17:1 to the asynchronous modem resource 1/03 (interface async1/03, also known as line tty219).

# debug call rsvp-sync events

To display events that occur during Resource Reservation Protocol (RSVP) setup, use the **debug call rsvp-sync events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call rsvp-sync events**

**no debug call rsvp-sync events**

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

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(3)XI1	This command was introduced.
	12.1(5)T	This command was integrated into Cisco IOS Release 12.1(5)T.
	12.2(2)XB1	This command was implemented on the Cisco AS5850.
	12.2(11)T	Support for the command was implemented in Cisco AS5850 images.

**Usage Guidelines** It is highly recommended that you log the output from the **debug call rsvp-sync events** command to a buffer, rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

**Examples** The following example shows a portion of sample output for a call initiating RSVP when using the **debug call rsvp-sync events** command:

```
00:03:25: Parameters: localip: 10.19.101.117 :localport: 16660
00:03:25: Parameters: remoteip: 10.19.101.116 :remoteport: 17568
00:03:25: QoS Primitive Event for Call id 0x1 : QoS Listen
00:03:25: Lookup to be done on hashkey 0x1 in hash table 0x61FC2498
00:03:25: Hashed entry 0x1 in call table 0x61FC2498
00:03:25: Entry Not found
00:03:25: Parameters: localip: 10.19.101.117
00:03:25:         remoteip: 10.19.101.116
00:03:25: QoSpcb : 0x61FC34D8
```

```

00:03:25: Response Status : 0
Starting timer for call with CallId 0x1 for 10000 secs

00:03:25: Handling QoS Primitive QoS Listen

00:03:25: Establishing RSVP RESV state : rsvp_request_reservation()

00:03:25: For streams from 10.19.101.116:17568 to 10.19.101.117:16660

00:03:25: RSVP Confirmation required

00:03:25: QoS Primitive Event for Call id 0x1 : QoS Resv
00:03:25: Lookup to be done on hashkey 0x1 in hash table 0x61FC2498

00:03:25: Hashed entry 0x1 in call table 0x61FC2498

00:03:25: Initiating RVSP PATH messages to be Sent : reg_invoke_rsvp_advertise_sender()

00:03:25: Advertizing for streams to 10.19.101.116:17568 from 10.19.101.117:16660

00:03:25: RESV notification event received is : 2

00:03:25: Received RESVCONFIRM

00:03:25: RESV CONFIRM message received from 10.19.101.116 for RESV setup from
10.19.101.117

00:03:25: RESV event received is : 0

00:03:25: RESV message received from 10.19.101.116:17568 for streams from
10.19.101.117:16660

00:03:25: RESERVATIONS ESTABLISHED : CallId: 1Stop timer and notify Session Protocol of
Success (ie. if notification requested)

00:03:25: Invoking spQoSresvCallback with Success

```

**Related Commands**

Command	Description
<b>call rsvp-sync</b>	Enables synchronization between RSVP and the H.323 voice signaling protocol.
<b>call rsvp-sync resv-timer</b>	Sets the timer for RSVP reservation setup.
<b>debug call rsvp-sync func-trace</b>	Displays messages about the software functions called by RSVP synchronization.
<b>show call rsvp-sync conf</b>	Displays the RSVP synchronization configuration.
<b>show call rsvp-sync stats</b>	Displays statistics for calls that attempted RSVP reservation.

# debug call rsvp-sync func-trace

To display messages about software functions called by Resource Reservation Protocol (RSVP), use the **debug call rsvp-sync func-trace** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call rsvp-sync func-trace**

**no debug call rsvp-sync func-trace**

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

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(3)XI1	This command was introduced.
	12.1(5)T	This command was integrated into Cisco IOS Release 12.1(5)T.
	12.2(2)XB1	This command was implemented on the Cisco AS5850.

**Usage Guidelines** It is highly recommended that you log the output from the **debug call rsvp-sync func-trace** command to a buffer, rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

**Examples** The following example shows a portion of sample output for a call initiating RSVP when using the **debug call rsvp-sync func-trace** command in conjunction with the **debug call rsvp-sync events** command:

```
00:03:41: Entering Function QoS_Listen
00:03:41: Parameters:localip:10.10.101.116 :localport:17568
00:03:41:remoteip:10.10.101.117 :remoteport:0
00:03:41: Entering Function qos_dequeue_event
00:03:41: Entering Function process_queue_event
00:03:41: QoS Primitive Event for Call id 0x2 :QoS Listen
00:03:41: Entering Function get_pcb
00:03:41: Entering Function hash_tbl_lookup
00:03:41:Lookup to be done on hashkey 0x2 in hash table 0x61FAECD8
```

```

00:03:41: Entering Function hash_func
00:03:41:Hashed entry 0x2 in call table 0x61FAECD8
00:03:41:Entry Not found
00:03:41: Entering Function qos_dequeue_pcb
00:03:41: Entering Function qos_initialize_pcb
00:03:41: Parameters:localip:10.10.101.116
00:03:41:  remoteip:10.10.101.117
00:03:41: QoSpcb :0x61FAFD18
00:03:41: Response Status :0
00:03:41: Entering Function hash_tbl_insert_entry
00:03:41: Entering Function hash_func
00:03:41: Handling QoS Primitive QoS Listen
00:03:41: Entering Function qos_dequeue_hash_port_entry
00:03:41: Entering Function qos_port_tbl_insert_entry
00:03:41: Entering Function hash_func
00:03:41: Doing RSVP Listen :rsvp_add_ip_listen_api()

```

**Related Commands**

Command	Description
<b>call rsvp-sync</b>	Enables synchronization between RSVP and the H.323 voice signaling protocol.
<b>call rsvp-sync resv-timer</b>	Sets the timer for RSVP reservation setup.
<b>debug call rsvp-sync events</b>	Displays the events that occur during RSVP synchronization.
<b>show call rsvp-sync conf</b>	Displays the RSVP synchronization configuration.
<b>show call rsvp-sync stats</b>	Displays statistics for calls that attempted RSVP reservation.

# debug call threshold

To see details of the trigger actions, use the **debug call threshold** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call threshold** *module*

**no debug call threshold**

<b>Syntax Description</b>	<i>module</i>	The <i>module</i> argument can be one of the following: <ul style="list-style-type: none"> <li>• <b>core</b>—Traces the resource information.</li> <li>• <b>detail</b>—Traces for detail information.</li> </ul>
---------------------------	---------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>Defaults</b>	Disabled
-----------------	----------

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

Command History	Release	Modification
	12.2(2)XA	This command was introduced.
	12.2(4)T	The command was integrated into Cisco IOS Release 12.2(4)T. Support for the Cisco AS5300, Cisco AS5350, and Cisco AS5400 is not included in this release.
	12.2(2)XB1	This command was implemented on the Cisco AS5850 platform.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on Cisco 7200 series routers.
	12.2(11)T	Support for this command was implemented on Cisco AS5850, Cisco AS5800, Cisco AS5300, Cisco AS5350, and Cisco AS5400 series images.

**Examples** The following is sample output from the **debug call threshold core** command:

```
Router# debug call threshold core
```

```
RSCCAC Core info debugging is on
```

The following is sample output from the **debug call threshold detail** command:

```
Router# debug call threshold detail
```

```
All RSCCAC info debugging is on
```

# debug call treatment action

To debug the call treatment actions, use the **debug call treatment action** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug call treatment action**

**no debug call treatment action**

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

**Defaults** Disabled

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.2(2)XA	This command was introduced.
	12.2(4)T	The command was integrated into Cisco IOS Release 12.2(4)T.
	12.2(2)XB1	This command was implemented on the Cisco AS5850 platform.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on Cisco 7200 series routers.
	12.2(11)T	Support for this command was implemented on Cisco AS5850, Cisco AS5800, Cisco AS5300, Cisco AS5350, and Cisco AS5400 series images.

**Examples** Debug actions are performed on calls by call treatment. The following sample output shows that call treatment is turned on:

```
Router# debug call treatment action

Call treatment action debugging is on
```

# debug cas

To debug channel-associated signaling (CAS) messages and to debug the establishment of a time-division multiplexing (TDM) connection between a DS0 and a digital modem, use the **debug cas** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug cas slot** *slot number* **port** *port number*

**no debug cas slot** *slot number* **port** *port number*

Syntax Description	slot <i>slot number</i>	Slot and slot number. Valid values are 0 and 1.
	port <i>port number</i>	Port and port number. Valid values are 0 and 1.

**Defaults** Disabled

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(7)T	This command was introduced for the Cisco AS5200 and AS5300 platforms.
	12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T and support was added for the Cisco 2600 series and Cisco 3600 series platforms.
	12.3(1)	This command was integrated into Cisco IOS Release 12.3(1) and support was added for the Cisco 2600 XM series, Cisco 2691, and Cisco 3700 series platforms.
	12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.

**Usage Guidelines** When the NM-xCE1T1PRI network module is used with an NM-xDM and a DS0-group is configured under the controller, you can use the **debug cas** command to debug CAS signaling messages and the establishment of a TDM connection between a DS0 and a digital modem. Use the **debug cas** command to identify and troubleshoot call connection problems on a T1/E1 interface. With this command, you can trace the complete sequence of incoming and outgoing calls.

**Examples** The following shows an example session to enable debugging CAS and generate troubleshooting output:

```
Router# show debug
Router# debug cas slot 1 port 0

CAS debugging is on
Router#
debug-cas is on at slot(1) dsx1(0)

Router# show debug

CAS debugging is on
```

The following example shows output for the first outgoing call:

```
Router# p 1.1.1.2
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.2, timeout is 2 seconds:
*Mar 2 00:17:45: dsx1_alloc_cas_channel: channel 0 dsx1_timeslot
1(0/0): TX SEIZURE (ABCD=0001)(0/0): RX SEIZURE_ACK (ABCD=1101)(0/1):
RX_IDLE (ABCD=1001)(0/2): RX_IDLE (ABCD=1001)(0/3): RX_IDLE
(ABCD=1001)(0/4): RX_IDLE (ABCD=1001)(0/5): RX_IDLE (ABCD=1001)(0/6):
RX_IDLE (ABCD=1001)(0/7): RX_IDLE (ABCD=1001)(0/8): RX_IDLE
(ABCD=1001)(0/9): RX_IDLE (ABCD=1001)(0/10): RX_IDLE (ABCD=1001)(0/11):
RX_IDLE (ABCD=1001)(0/12): RX_IDLE (ABCD=1001)(0/13): RX_IDLE
(ABCD=1001)(0/14): RX_IDLE (ABCD=1001)(0/16): RX_IDLE (ABCD=1001)(0/17):
RX_IDLE (ABCD=1001)(0/18): RX_IDLE (ABCD=1001)(0/19): RX_IDLE
(ABCD=1001)(0/20): RX_IDLE (ABCD=1001)(0/21): RX_IDLE
(ABCD=1001)(0/22): RX_IDLE (ABCD=1001)(0/23): RX_IDLE
(ABCD=1001)(0/24): RX_IDLE (ABCD=1001)(0/25): RX_IDLE (ABCD=1001)(0/26):
RX_IDLE (ABCD=1001)(0/27): RX_IDLE (ABCD=1001)(0/28): RX_IDLE
(ABCD=1001)(0/29): RX_IDLE (ABCD=1001)(0/30): RX_IDLE
(ABCD=1001)...(0/0): RX ANSWERED (ABCD=0101).
Success rate is 0 percent (0/5)
```

```
Router#
```

```
*Mar 2 00:18:13.333: %LINK-3-UPDOWN: Interface Async94, changed state to up
*Mar 2 00:18:13.333: %DIALER-6-BIND: Interface As94 bound to profile Di1
*Mar 2 00:18:14.577: %LINEPROTO-5-UPDOWN: Line protocol on Interface Async94, changed
state to up
```

```
Router# p 1.1.1.2
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 160/180/236 ms
```

The following example shows that the call is cleared on the router:

```
Router# clear int dialer 1
```

```
Router#
```

```
(0/0): TX IDLE (ABCD=1001)(0/0): RX IDLE (ABCD=1001)
*Mar 2 00:18:28.617: %LINK-5-CHANGED: Interface Async94, changed state to reset
*Mar 2 00:18:28.617: %DIALER-6-UNBIND: Interface As94 unbound from profile Di1
*Mar 2 00:18:29.617: %LINEPROTO-5-UPDOWN: Line protocol on Interface Async94, changed
state to down
et2-c3745-1#
*Mar 2 00:18:33.617: %LINK-3-UPDOWN: Interface Async94, changed state to down
```

The following example shows a subsequent outbound CAS call:

```
Router# p 1.1.1.2
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.2, timeout is 2 seconds:
*Mar 2 00:18:40: dsx1_alloc_cas_channel: channel 5 dsx1_timeslot
6(0/5): TX SEIZURE (ABCD=0001)(0/5): RX SEIZURE_ACK
(ABCD=1101)...(0/5): RX ANSWERED (ABCD=0101).
Success rate is 0 percent (0/5)
```

```
Router#
```

```
*Mar 2 00:19:08.841: %LINK-3-UPDOWN: Interface Async93, changed state to up
*Mar 2 00:19:08.841: %DIALER-6-BIND: Interface As93 bound to profile Di1
*Mar 2 00:19:10.033: %LINEPROTO-5-UPDOWN: Line protocol on Interface Async93, changed
state to up
```

```
Router# p 1.1.1.2
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 160/167/176
ms
```

The following example shows the call cleared by the switch:

```
Router#
(0/5): TX IDLE (ABCD=1001)(0/5): RX IDLE (ABCD=1001)
*Mar 2 00:19:26.249: %LINK-5-CHANGED: Interface Async93, changed state to reset
*Mar 2 00:19:26.249: %DIALER-6-UNBIND: Interface As93 unbound from profile Dil
*Mar 2 00:19:27.249: %LINEPROTO-5-UPDOWN: Line protocol on Interface Async93, changed
state to down
Router#
*Mar 2 00:19:31.249: %LINK-3-UPDOWN: Interface Async93, changed state to down
```

The following example shows an incoming CAS call:

```
Router#
(0/0): RX SEIZURE (ABCD=0001)
*Mar 2 00:22:40: dsx1_alloc_cas_channel: channel 0 dsx1_timeslot
1(0/0): TX SEIZURE_ACK (ABCD=1101)(0/0): TX ANSWERED (ABCD=0101)
Router#
*Mar 2 00:23:06.249: %LINK-3-UPDOWN: Interface Async83, changed state to up
*Mar 2 00:23:06.249: %DIALER-6-BIND: Interface As83 bound to profile Dil
*Mar 2 00:23:07.653: %LINEPROTO-5-UPDOWN: Line protocol on Interface Async83, changed
state to up
```

## Related Commands

Command	Description
<b>show debug</b>	Displays information about the types of debugging that are enabled for your router.

# debug ccaal2 session

To display the ccaal2 function calls during call setup and teardown, use the **debug ccaal2 session** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ccaal2 session**

**no debug ccaal2 session**

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

**Defaults** Debugging for ATM Adaptation Layer type 2 (AAL2) sessions is not enabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.1(1)XA	This command was introduced for the Cisco MC3810 series.
	12.1(2)T	This command was integrated in Cisco IOS Release 12.1(2)T.
	12.2(2)T	Support for this command was implemented on the Cisco 7200 series routers.

**Usage Guidelines** Use this command when troubleshooting an AAL2 trunk setup or teardown problem.

**Examples** The following example shows sample output from the **debug ccaal2 session** command for a forced shutdown of a voice port:

```
Router# debug ccaal2 session

CCAAL2 Session debugging is on
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# voice-port 2/0:0
Router(config-voiceport)# shutdown

00:32:45:ccaal2_call_disconnect:peer tag 0
00:32:45:ccaal2_evhandle_call_disconnect:Entered
00:32:45:ccaal2_call_cleanup:freeccb 1, call_disconnected 1
00:32:45:starting incoming timer:Setting accept_incoming to FALSE and
00:32:45:timer 2:(0x622F6270)starts - delay (70000)
00:32:45:ccaal2_call_cleanup:Generating Call record
00:32:45:cause=81 tcause=81 cause_text=unspecified
00:32:45:ccaal2_call_cleanup:ccb 0x63FF1700, vdbPtr 0x62DFF2E0
freeccb_flag=1, call_disconnected_flag=1
00:32:45:%LINK-3-UPDOWN:Interface recEive and transMit2/0:0(1),
changed state to Administrative Shutdown
```

The following example shows sample output from the **debug ccaal2 session** command for a trunk setup on a voice port:

```
Router# debug ccaal2 session

Router(config-voiceport)# no shutdown
Router(config-voiceport)#
00:35:28:%LINK-3-UPDOWN:Interface recEive and transMit2/0:0(1),
changed state to up
00:35:35:ccaal2_call_setup_request:Entered
00:35:35:ccaal2_evhandle_call_setup_request:Entered
00:35:35:ccaal2_initialize_ccb:preferred_codec set (-1) (0)
00:35:35:ccaal2_evhandle_call_setup_request:preferred_codec
set (5) (40). VAD is 1
00:35:35:ccaal2_call_setup_trunk:subchannel linking
successfulccaal2_receive:xmitFunc is NULL

00:35:35:ccaal2_caps_ind:PeerTag = 49
00:35:35:      codec(preferred) = 1, fax_rate = 2, vad = 2
00:35:35:      cid = 56, config_bitmask = 258, codec_bytes = 40,
          signal_type=8
00:35:36:%HTSP-5-UPDOWN:Trunk port(channel) [2/0:0(1)] is up
Router(config-voiceport)#
```

---

**Related Commands**

Command	Description
<b>show debug</b>	Shows which debug commands are enabled.

---

# debug ccfrf11 session

To display the ccfrf11 function calls during call setup and teardown, use the **debug ccfrf11 session** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ccfrf11 session**

**no debug ccfrf11 session**

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

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(3)XG	This command was introduced for the Cisco 2600 and Cisco 3600 series routers.
	12.0(4)T	This command was integrated into Cisco IOS Release 12.0(4)T.
	12.0(7)XK	This command was first supported on the Cisco MC3810 series.
	12.1(2)T	Support for this command was implemented in Cisco MC3810 images.

**Usage Guidelines** Use this command to display debug information about the various FRF.11 VoFR service provider interface (SPI) functions. Note that this debug command does not display any information regarding the proprietary Cisco switched-VoFR SPI.

This debug is useful only when the session protocol is “frf11-trunk.”

**Examples** The following is sample output from the **debug ccfrf11 session** command:

```
Router# debug ccfrf11 session

INCOMING CALL SETUP (port setup for answer-mode):
*Mar 6 18:04:07.693:ccfrf11_process_timers:scb (0x60EB6040) timer (0x60EB6098) expired
*Mar 6 18:04:07.693:Setting accept_incoming to TRUE
*Mar 6 18:04:11.213:ccfrf11_incoming_request:peer tag 800:callingNumber=+2602100,
calledNumber=+3622110
*Mar 6 18:04:11.213:ccfrf11_initialize_ccb:preffered_codec set(-1)(0)
*Mar 6 18:04:11.213:ccfrf11_evhandle_incoming_call_setup_request:calling +2602100,
called +3622110 Incoming Tag 800
*Mar 6 18:04:11.217:ccfrf11_caps_ind:PeerTag = 800
*Mar 6 18:04:11.217:      codec(preferred) = 4, fax_rate = 2, vad = 2
*Mar 6 18:04:11.217:      cid = 30, config_bitmask = 0, codec_bytes = 20, signal_type=2
*Mar 6 18:04:11.217:      required_bandwidth 8192
*Mar 6 18:04:11.217:ccfrf11_caps_ind:Bandwidth reservation of 8192 bytes succeeded.
*Mar 6 18:04:11.221:ccfrf11_evhandle_call_connect:Entered

CALL SETUP (MASTER):
5d22h:ccfrf11_call_setup_request:Entered
5d22h:ccfrf11_evhandle_call_setup_request:Entered
5d22h:ccfrf11_initialize_ccb:preffered_codec set(-1)(0)
```

```

5d22h:ccfrr11_evhandle_call_setup_request:preffered_codec set(9) (24)
5d22h:ccfrr11_call_setup_trunk:subchannel linking successful
5d22h:ccfrr11_caps_ind:PeerTag = 810
5d22h:      codec(preferred) = 512, fax_rate = 2, vad = 2
5d22h:      cid = 30, config_bitmask = 1, codec_bytes = 24, signal_type=2
5d22h:      required_bandwidth 6500
5d22h:ccfrr11_caps_ind:Bandwidth reservation of 6500 bytes succeeded.

```

## CALL TEARDOWN:

```

*Mar 6 18:09:14.805:ccfrr11_call_disconnect:peer tag 0
*Mar 6 18:09:14.805:ccfrr11_evhandle_call_disconnect:Entered
*Mar 6 18:09:14.805:ccfrr11_call_cleanup:freeccb 1, call_disconnected 1
*Mar 6 18:09:14.805:ccfrr11_call_cleanup:Setting accept_incoming to FALSE and starting
incoming timer
*Mar 6 18:09:14.809:timer 2:(0x60EB6098)starts - delay (70000)
*Mar 6 18:09:14.809:ccfrr11_call_cleanup:Alive timer stopped
*Mar 6 18:09:14.809:timer 1:(0x60F64104) stops
*Mar 6 18:09:14.809:ccfrr11_call_cleanup:Generating Call record
*Mar 6 18:09:14.809:cause=10 tcause=10 cause_text="normal call clearing."
*Mar 6 18:09:14.809:ccfrr11_call_cleanup:Releasing 8192 bytes of reserved bandwidth
*Mar 6 18:09:14.809:ccfrr11_call_cleanup:ccb 0x60F6404C, vdbPtr 0x610DB7A4
freeccb_flag=1, call_disconnected_flag=1

```

## Related Commands

Command	Description
<b>debug call rsvp-sync events</b>	Displays the ccsvoice function calls during call setup and teardown.
<b>debug ccsvoice vofr-session</b>	Displays the ccsvoice function calls during call setup and teardown.
<b>debug vtsp session</b>	Displays the first 10 bytes (including header) of selected VoFR subframes for the interface.