Understanding debug atm event Output on ATM Router Interfaces

Document ID: 10437

Contents

Introduction

Multiple processors residing on a dedicated system processor module as well as locally on interface hardware work together to ensure successful transmission and receipt of packets over ATM virtual circuits (VCs). These processors communicate among themselves by posting messages to perform such functions as VC setup and teardown, physical−layer statistics collection, and alarm generation. These messages, called love letters or love messages, are written by one processor into a block of memory. A receiving processor then reads the message. The output of the **debug atm events** command provides a window into this messaging mechanism, such as the following output from a PA−A3.

```
Jun 17 12:48:50.631 BST: atmdx_mailbox_proc(ATM5/0/0): received report type 2
Jun 17 12:48:50.631 BST: atmdx_process_love_letter(ATM5/0/0): 2 VCs core
 statistics
Jun 17 12:48:55.631 BST: atmdx_mailbox_proc(ATM5/0/0): received report type 3
Jun 17 12:48:55.631 BST: atmdx_process_love_letter(ATM5/0/0): 1 VCs aux
 statistics
```

The purpose of this document is illustrate sample **debug atm event** output to help distinguish between informational messages and messages that point to an operational problem. This document also reviews standard ATM interface software architecture.

Caution: Before issuing debug commands, please refer to Important Information on Debug Commands. The **debug atm events** command may print a large amount of disruptive debug output on a production router depending on the number of VCs for which it needs to report statistics as well as the amount of VC−related events.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

This document is not restricted to specific software and hardware versions.

Conventions

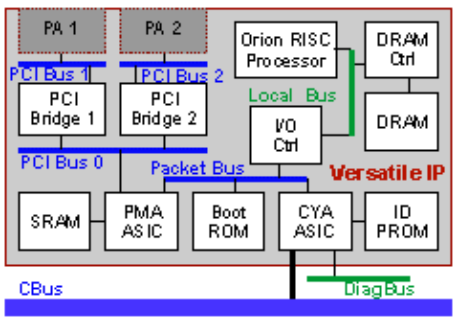
For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Understanding Software Functional Blocks

All ATM interfaces use a software architecture that consists of multiple blocks. Before we walk through these software blocks, we first need to understand Cisco IOS® Software drivers and the PCI bus architecture inside your router.

A driver allows software engineers to implement something called hardware abstraction. It allows engineers to create a fundamental set of software blocks that run on any platform, and then use drivers to adapt this platform−independent code to a specific platform such as the 7200 series or the 3600 series.

The PA−A3 supports a PCI host driver that allows the Segmentation and Reassembly's (SAR's) processor to interface with the peripheral component interconnect (PCI) buses that run length of the 7200/7400 series, as well as the versatile interface processor (VIP) on RSP platforms. PCI buses serve as a data path between port adapters and host memory on the VIP or on the Network Processing Engine (NPE)/ Network Services Engine (NSE). The following diagram illustrates the architecture of the VIP2 and the location of the PCI buses:



This table lists the software blocks on the PA−A3:

Software Block	Function
ATM core	Platform− or PA−independent software functions that all ATM interfaces use. For example, ATM core handles OAM and ILMI management.
Platform driver	Platform−dependent software functions that "bridge" the general ATM core software with the PCI host driver software. ATM core and the PCI host driver exchange commands, status updates, and statistics via the bridge. The platform ATM driver also handles receive−packet forwarding, platform−specific initialization functions, and physical−layer statistics as shown in the **show controller atm** display.

PCI host driver	Provides the PCI host interface for the SAR chip on the PA−A3. Performs several key functions: • Downloads firmware to the SAR • Transports packets • Collects statistics • Monitors framer alarms
Host interface	Part of each SAR's hardware functional block. Performs several key actions: • Downloads boot code to configure the SARs and enables them to exchange control data with the PCI host driver. • Generates interrupts when the SAR needs to write cells into memory on the receive path and schedule cells on the transmit path. • Returns empty buffers to the PCI host driver. • Processes commands sent from the PCI host driver and relays locally collected statistics to the PCI host driver.
Firmware	Start−up or boot code as well as optimized runtime images for the ATM processor unit (APU) on the receive and transmit SARs. Downloaded from the PCI host driver.

On the RSP/VIP platform, the platform driver resides in the RSP system image and VIP system image, while the PCI host driver is part of the VIP system image. On the 7200 platform, both drivers are part of the system image.

The PA−A3−specific software is bundled with the VIP software or with the system software for other supporting platforms.

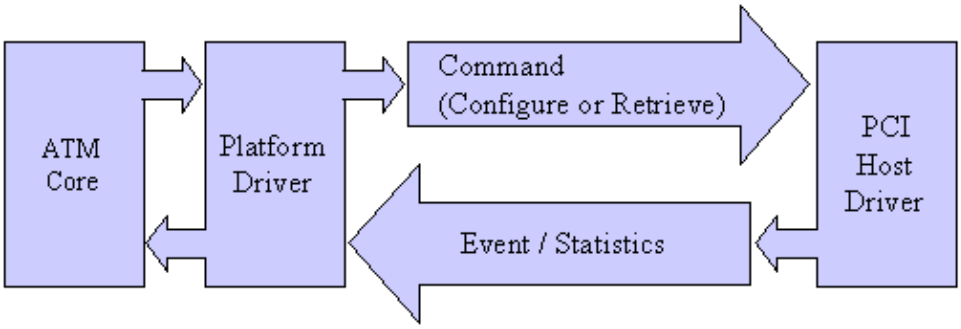
What Is a Mailbox?

As noted above, a mailbox is part of a messaging model that Cisco IOS uses to transport messages between two CPUs. Here is how this process generally works:

1. A driver allocates a message buffer.
2. A love note or letter fills the message buffer.
3. The receiving processor reads the message buffer.
4. When finished reading the command buffer, the processor generates a "message done" interrupt.
5. The message buffer is returned to the free buffer pool.

Now this document examines two sets of messages exchanged between processors running the Cisco IOS Software components described in the table above.

ATM Core to Platform Driver and PCI Host Driver



The PCI host driver collects per−VC statistics on each packet. The VIP platform driver autonomously relays these statistics to the RSP platform driver via a love note every second. The **show atm vc** command displays the current VC data. The VIP platform driver relays framer statistics to the RSP every 10 seconds. When the system initializes, it creates a special background process that handles the autonomous statistics from the VIP as a scheduled process rather than at the interrupt level to minimize system interruption.

The **debug atm events** command prints output on VC−related events such as setup and teardown.

Function	Description
setupvc	Set up a VC. The platform−dependent driver delivers the request to the PCI host driver.
teardownvc	Tears down an existing VC. The platform−dependent driver relays the request to the PCI host driver.
getvc_stats	Retrieves VC statistics on demand; supports only a single VC request.
qos_params_verify	Verifies QoS paramters before a VC is set up.

PCI Host Driver to PA Firmware

The SAR internally consists of hardware functional blocks. One such block is the ATM processing unit (APU), which is a miniRISC with customized logic for ATM−specific extensions. The PCI host driver and the APU, which runs the ATM firmware, communicate via a messaging mailbox. At any given time, one outstanding command for each APU is used to instruct the PA firmware to perform a specific task, such as a VC setup. The firmware relays per−VC and per−PA statistics to the PCI host driver every 10 seconds if the data changes.

The following output generated from **debug atm event** shows the commands sent by the PCI host driver to the firmware. The firmware returns only acknowledgments to indicate the success of the command. These acknowledgments are not displayed in the debug output.

```
7200−1.3(config)# int atm 6/0
7200−1.3(config−if)# pvc 1/100
7200−1.3(config−if−atm−vc)# vbr−nrt 45000 45000
7200−1.3#
17:07:43: atmdx_setup_vc(ATM6/0): vc:14 vpi:1 vci:100 state:2 config_status:0
17:07:43: atmdx_pas_vc_setup(ATM6/0): vcd 14, atm hdr 0x00100640, mtu 4482
17:07:43: VBR: pcr 96000, scr 96000, mbs 94
17:07:43:  vc tx_limit=1600, rx_limit=480
17:07:43:  Created 64−bit VC counterss
```

```
7200−1.3(config)# int atm 6/0
7200−1.3(config−if)# no pvc 1/100
7200−1.3(config−if)#
17:08:48: atmdx_teardown_vc(ATM6/0): idb state 4 vcd 14 state 4
17:08:48: atmdx_pas_teardown_vc(ATM6/0): vcd 14
```

Software Architecture of the IMA Network Module

Now this document applies the preceding information by walking through the software architecture of the inverse multiplexing over ATM (IMA) network module (NM) for the 2600 and 3600 router series.

The IMA NM has a "host" side to indicate functions or memory on the processor module and a "local" side to indicate functions or memory on the network module itself. The host side runs platform−independent and platform−dependent drivers. The local side executes firmware downloaded by the host drivers to the NM's onboard CPU. This image handles the physical−layer functions, including control of the framer ASIC, collection of physical−layer statistics, and generation of loopbacks and alarms. The Cisco IOS drivers and the NM firmware communicate via mail messages.

On the local side, the NM IMA also runs an IMA driver that similarly uses a message mailbox to communicate to the local CPU.

Messages in the direction of host side to local side are designed mostly for configuration. These messages include:

- Physical layer E1/T1 configuration data
- IMA group configuration
- Loopback configuration
- Debug configuration
- Query for IMA group/link status
- Query for RFC 1406 management information base (MIB) data
- Query for IMA MIB data

Messages sent in the direction of local side to host side are used to communicate line state changes and performance statistics, including these:

- Physical layer E1/T1 status changes
- IMA group status changes
- IMA link status changes
- Loopback status changes
- Debug messages
- Response of RFC 1406 MIB data
- Response of IMA MIB data

The following sample output illustrates the love notes used to setup and teardown a VC. We shut and no shut the physical interface to force the teardown. Note that "rs8234" refers to the SAR on the NM.

```
3640−1.1(config)# int atm2/ima2
3640−1.1(config−if)# pvc 1/1
3640−1.1(config−if−atm−vc)# shut
3640−1.1(config−if)#
*Mar  1 00:17:20.323:  Reserved bw for 1/1 Available bw = 6000
*Mar  1 00:17:20.323: rs8234_setup_vc(ATM2/IMA2): vc:4 vpi:1 vci:1
*Mar  1 00:17:20.323: rs8234_setup_vc_common() VCD=260 vp/vc=17/1 etype=0
*Mar  1 00:17:20.323: rs8234_setup_cos(ATM2/IMA2): vc:4 wred_name:− max_q:0
*Mar  1 00:17:20.327:  Created 64−bit VC counters
*Mar  1 00:17:20.327: rs8234_teardown_vc(ATM2/IMA2): vc:260 vpi:1 vci:1
```

```
*Mar  1 00:17:20.327: rs8234_teardown_vc proceeds (ATM2/IMA2): vc:260 vpi:1
 vci:1
*Mar  1 00:17:20.327: Status and ptr is 400 Status Q is 1
*Mar  1 00:17:20.331: Resetting ATM2/IMA2
*Mar  1 00:17:20.331: rs8234_teardown_vc(ATM2/IMA2): vc:260 vpi:1 vci:1
*Mar  1 00:17:20.331: rs8234_teardown_vc proceeds (ATM2/IMA2): vc:260 vpi:1 vci:1
*Mar  1 00:17:20.331:  Remove link with ports 8,links 4,channel 1
*Mar  1 00:17:22.327: %LINK-5-CHANGED: Interface ATM2/IMA2, changed state to administrativ
3640-1.1(config-if)# no shut
3640-1.1(config-if)#
*Mar  1 00:17:31.287: Resetting ATM2/IMA2
*Mar  1 00:17:31.287: IMA config_interface ATM2/IMA2
*Mar  1 00:17:31.287: IMA config_restart ATM2/IMA2
*Mar  1 00:17:31.287: IMA restarting 0 VCs
*Mar  1 00:17:31.287: rs8234_setup_vc(ATM2/IMA2): vc:4 vpi:1 vci:1
*Mar  1 00:17:31.287: rs8234_setup_vc_common() VCD=260 vp/vc=17/1 etype=0
*Mar  1 00:17:31.287: rs8234_setup_cos(ATM2/IMA2): vc:4 wred_name:- max_q:0
```

Related Information

- **Cisco ATM Port Adapter**
- **ATM Technology Support**
- **Technical Support − Cisco Systems**

Contacts & Feedback | Help | Site Map
© 2014 − 2015 Cisco Systems, Inc. All rights reserved. Terms & Conditions | Privacy Statement | Cookie Policy | Trademarks of Cisco Systems, Inc.

Updated: Nov 15, 2007 Document ID: 10437