VadaTech MCH and AMC229

10 GbE Switch Management CPU Console

February 25, 2010

Version 3.3.0

vadatech inc

THE POWER OF VISION

Notice

While reasonable efforts have been made to assure the accuracy of this document, VadaTech, Inc. assumes no liability resulting from any omissions in this document or from the use of the information obtained herein. VadaTech reserves the right to revise this document and to make changes periodically and the content hereof without obligation of VadaTech to notify any person of such revision or changes.

Electronic versions of this material may be read online, downloaded for personal use, or referenced in another document as a URL to the VadaTech Incorporated Web site. The text itself may not be published commercially in print or electronic form, edited, translated, or otherwise altered without the permission of VadaTech, Inc.

It is possible that this publication may contain reference to or information about VadaTech products (machines and programs), programming, or services that are not available in your country. Such references or information must not be construed to mean that VadaTech intends to announce such products, programming, or services in your country.

Trademarks

The VadaTech, Inc name and logo are registered trademarks of VadaTech Incorporated in the U.S.A. All other product or service names mentioned in this document are the property of their respective owners.

Revision History

Doc Rev	Description of Change	Revision Date
1.0.0	Initial version	11/12/2008
2.0.0	Added instructions for setting the IP addresses. Added instructions for using the Fujitsu reference software (axdrvd and axcli). Added instructions for configuring IP routing. Added instructions for configuring Ethernet bridging and filtering.	03/18/2009
2.1.0	Corrected serial port name in diagram.	03/18/2009
3.0.0	Added description of 10GbE switch on UTC002 (DA116) board variant in addition to the original UTC001 (DA112).	08/17/2009
3.0.1	Minor corrections and additional information.	09/2/2009
3.1.0	New information for BSP 8 including support for the new AMC229 board as well as the new RSTP support for all boards.	12/4/2009
3.2.0	Added new 'axel_linkstat' command and minor improvements.	12/14/2009
3.3.0	Added description of VT851, VT852, VT853 (DA152) board variant.	2/25/2010

Table of Contents

Figures

Tables

1 Overview

1.1 UTC001 Hardware

The UTC001 MCH module includes various CPUs, one of which is the 10GbE Switch Management CPU. This CPU is found on the MCH on the DA112 daughter card when the 10GbE switch option is purchased. This CPU is responsible for managing the 10GbE switch and can also be used to create an Ethernet Bridge or an IP Route between the 10Gb and 1Gb Ethernet segments. The architecture of the DA112 daughter card is shown below. Note that the GbE connection to the 'UTC001 Base Board' connects to the 1GbE Managed Layer 2 switch found there. Please refer to the VadaTech UTC001 Hardware Reference Manual for a broader overview:

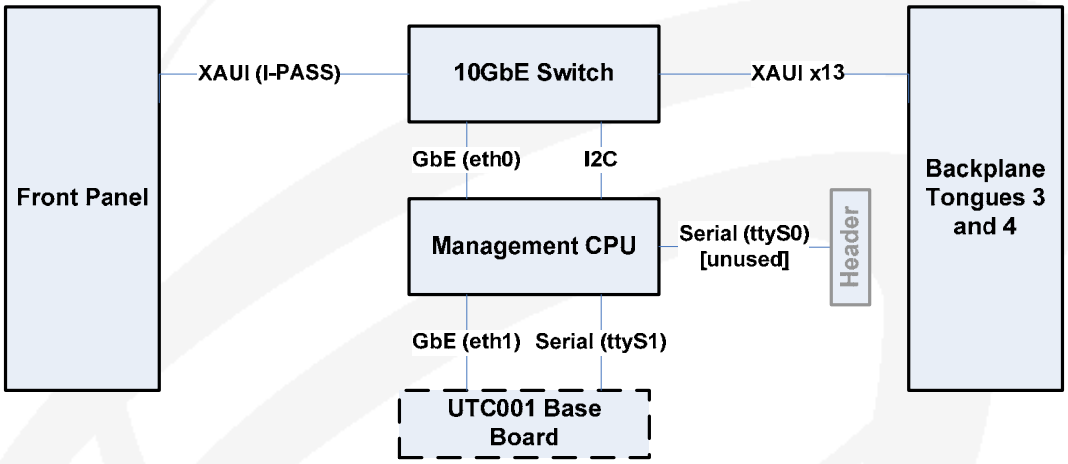

Figure 1: DA112 (UTC001) Block Diagram

Physical Port #	Port Description	Physical Port #	Port Description
0	AMC 3 (XAUI/1GbE)	10	(no connect)
1	(no connect)	11	AMC 8 (XAUI/1GbE)
2	(no connect)	12	AMC 11 (XAUI/1GbE)
3	AMC 9 (XAUI/1GbE)	13	AMC 6 (XAUI/1GbE)
4	AMC 10 (XAUI/1GbE)	14	Front I-PASS (XAUI/1GbE)
5	(no connect)	15	AMC 1 (XAUI/1GbE)
6	(no connect)	16	AMC 5 (XAUI/1GbE)
7	AMC 2 (XAUI/1GbE)	17	AMC 12 (XAUI/1GbE)
8	AMC 4 (XAUI/1GbE)	18	MCH <-> MCH (XAUI/1GbE)
9	Management CPU (1GbE – eth0)	19	AMC 7 (XAUI/1GbE)

Table 1: 10GBE Switch Port Configuration on DA112 (UTC001)

1.2 UTC002 Hardware

The UTC002 MCH module includes various CPUs, one of which is the 10GbE Switch Management CPU. This CPU is found on the MCH on the DA116 daughter card when the 10GbE switch option is purchased. This CPU is responsible for managing the 10GbE switch and can also be used to create an Ethernet Bridge or an IP Route between the 10Gb and 1Gb Ethernet segments. However, on this MCH the 10GbE switch already has a dedicated link to the 1GbE switch and therefore it is not necessary to create a bridge using the CPU. The architecture of the DA116 daughter card is shown below. Note that the two GbE connections to the 'UTC002 Base Board' connect to the 1GbE Managed Layer 2 switch found there. Please refer to the VadaTech UTC002 Hardware Reference Manual for a broader overview:

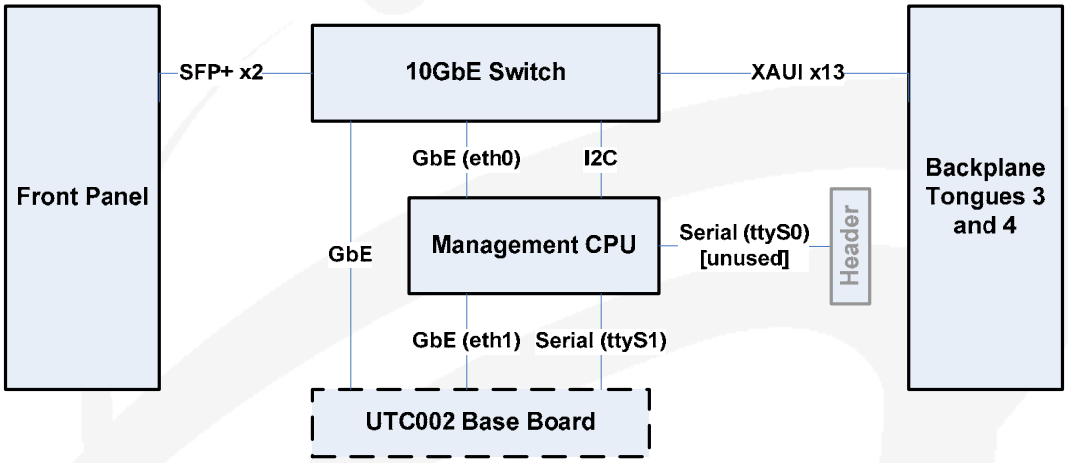

Figure 2: DA116 (UTC002) Block Diagram

Physical Port #	Port Description	Physical Port #	Port Description
0	AMC 3 (XAUI/1GbE)	10	Front SFP+ 1 (10GbE/1GbE auto SFP detect)
1	1G Switch-to-10G switch (1GbE)	11	AMC 8 (XAUI/1GbE)
2	(no connect)	12	AMC 11 (XAUI/1GbE)
3	AMC 9 (XAUI/1GbE)	13	AMC 6 (XAUI/1GbE)
4	AMC 10 (XAUI/1GbE)	14	Front SFP+ 0 (10GbE/1GbE auto SFP detect)
5	(no connect)	15	AMC 1 (XAUI/1GbE)
6	(no connect)	16	AMC 5 (XAUI/1GbE)
7	AMC 2 (XAUI/1GbE)	17	AMC 12 (XAUI/1GbE)
8	AMC 4 (XAUI/1GbE)	18	MCH <-> MCH (XAUI/1GbE)
9	Management CPU (1GbE – eth0)	19	AMC 7 (XAUI/1GbE)

Table 2: 10GBE Switch Port Configuration on DA116 (UTC002)

1.3 VT851/VT852/VT853 Hardware

The VT851/VT852/VT853 U1 chassis MCH block includes various CPUs, one of which is the 10GbE Switch Management CPU. This CPU is found on the DA152 daughter card when the 10GbE switch option is purchased. This CPU is responsible for managing the 10GbE switch and can also be used to create an Ethernet Bridge or an IP Route between the 10Gb and 1Gb Ethernet segments. However, on this MCH the 10GbE switch already has a dedicated link to the 1GbE switch and therefore it is not necessary to create a bridge using the CPU. The architecture of the DA152 daughter card is shown below. Note that the two GbE connections to the 'VT85x Base Board' connect to the 1GbE Managed Layer 2 switch found there. Please refer to the VadaTech VT851/VT852/VT853 Hardware Reference Manual for a broader overview.

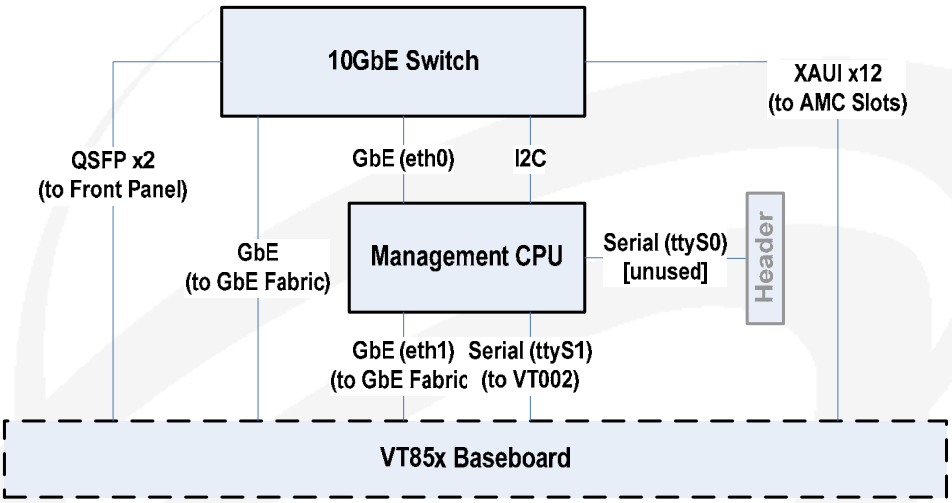

Figure 3: DA152 (VT851/VT852/VT853) Block Diagram

Physical Port #	Port Description	Physical Port #	Port Description
0	AMC 3 Ports 4-7 (XAUI/1GbE)	10	(no connect)
1	1G Switch-to-10G switch (1GbE)	11	AMC 8 Ports 4-7 (XAUI/1GbE)
2	(no connect)	12	AMC 11 Ports 4-7 (XAUI/1GbE)
3	AMC 9 Ports 4-7 (XAUI/1GbE)	13	AMC 6 Ports 4-7 (XAUI/1GbE)
4	AMC 10 Ports 4-7 (XAUI/1GbE)	14	(no connect)
5	(no connect)	15	AMC 1 Ports 4-7 (XAUI/1GbE)
6	(no connect)	16	AMC 5 Ports 4-7 (XAUI/1GbE)
7	AMC 2 Ports 4-7 (XAUI/1GbE)	17	AMC 12 Ports 4-7 (XAUI/1GbE)
8	AMC 4 Ports 4-7 (XAUI/1GbE)	18	(no connect)
9	Management CPU (1GbE – eth0)	19	AMC 7 Ports 4-7 (XAUI/1GbE)

Table 3: 10GBE Switch Port Configuration for VT851 chassis (DA152)

Physical Port #	Port Description	Physical Port #	Port Description
0	AMC 3 Ports 8-11 (XAUI/1GbE)	10	(no connect)
1	1G Switch-to-10G switch (1GbE)	11	AMC 2 Ports 4-7 (XAUI/1GbE)
2	(no connect)	12	AMC 5 Ports 8-11 (XAUI/1GbE)
3	AMC 3 Ports 4-7 (XAUI/1GbE)	13	AMC 6 Ports 8-11 (XAUI/1GbE)
4	AMC 4 Ports 8-11 (XAUI/1GbE)	14	Front QSFP 1 (XAUI/1GbE)
5	(no connect)	15	AMC 1 Ports 4-7 (XAUI/1GbE)
6	(no connect)	16	AMC 5 Ports 4-7 (XAUI/1GbE)
7	AMC 2 Ports 8-11 (XAUI/1GbE)	17	AMC 6 Ports 4-7 (XAUI/1GbE)
8	AMC 4 Ports 4-7 (XAUI/1GbE)	18	Front QSFP 0 (XAUI/1GbE)
9	Management CPU (1GbE – eth0)	19	AMC 1 Ports 8-11 (XAUI/1GbE)

Table 4: 10GBE Switch Port Configuration for VT852 chassis (DA152)

Physical Port #	Port Description	Physical Port #	Port Description
0	AMC 3 Ports 4-7 (XAUI/1GbE)	10	(no connect)
1	1G Switch-to-10G switch (1GbE)	11	(no connect)
2	(no connect)	12	(no connect)
3	(no connect)	13	AMC 6 Ports 4-7 (XAUI/1GbE)
4	(no connect)	14	Front QSFP 1 (XAUI/1GbE)
5	(no connect)	15	AMC 1 Ports 4-7 (XAUI/1GbE)
6	(no connect)	16	AMC 5 Ports 4-7 (XAUI/1GbE)
7	AMC 2 Ports 4-7 (XAUI/1GbE)	17	(no connect)
8	AMC 4 Ports 4-7 (XAUI/1GbE)	18	Front QSFP 0 (XAUI/1GbE)
9	Management CPU (1GbE – eth0)	19	(no connect)

Table 5: 10GBE Switch Port Configuration for VT853 chassis (DA152)

1.4 AMC229 Hardware

The AMC229 card includes a 10GbE switch and a Switch Management CPU. The switch includes 3 XAUI (10GbE or 1GbE) links to the backplane fabric as well as 2 GbE links to the backplane base channels. On the front panel it includes 8 SFP+ (10GbE or 1GbE) ports to the switch with auto-speed selection. The architecture of the AMC229 card is shown below.

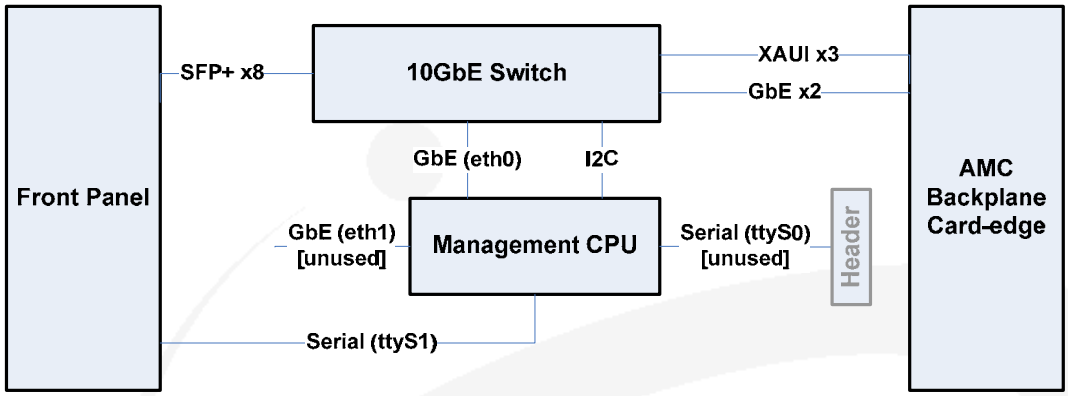

Figure 4: AMC229 Block Diagram

Physical Port #	Port Description	Physical Port #	Port Description
0	Front SFP+ 5 (10GbE/1GbE auto SFP detect)	10	(no connect)
1	(no connect)	11	Front SFP+ 3 (10GbE/1GbE auto SFP detect)
2	(no connect)	12	Front SFP+ 4 (10GbE/1GbE auto SFP detect)
3	Front SFP+ 6 (10GbE/1GbE auto SFP detect)	13	AMC base 0 (1GbE)
4	Front SFP+ 1 (10GbE/1GbE auto SFP detect)	14	AMC fabric 4-7 (XAUI/1GbE)
5	(no connect)	15	Front SFP+ 7 (10GbE/1GbE auto SFP detect)
6	(no connect)	16	AMC base 1 (1GbE)
7	Front SFP+ 2 (10GbE/1GbE auto SFP detect)	17	AMC fabric 17-20 (XAUI/1GbE)
8	Front SFP+ 0 (10GbE/1GbE auto SFP detect)	18	AMC Fabric 8-11 (XAUI/1GbE)
9	Management CPU (1GbE – eth0)	19	(no connect)

Table 6: 10GBE Switch Port Configuration on AMC229

NOTE: The SFP+ cages are numbered 0, 1, 2, 3 for row 1 and 4, 5, 6, 7 for row 2. SFP+ 4 is the one closest to the AMC handle (see diagram on next page).

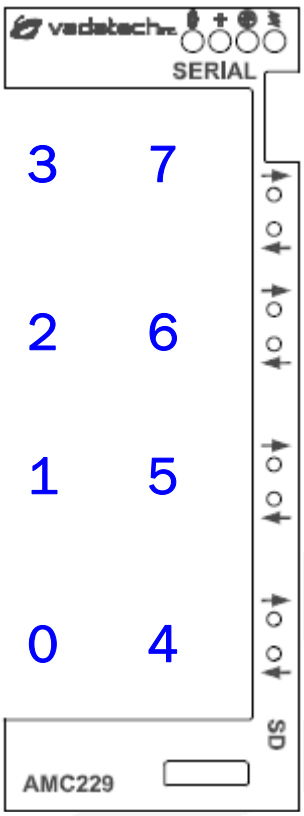

Figure 5: AMC229 SFP+ identification

SFP+ 0 is the only port available while in U-Boot and the speed is fixed at 1000Base-X. The other ports become available after the RSTP daemon starts up under Linux.

1.5 General

The remainder of this document will explain how to gain access to the DA112/DA116/DA152/AMC229 CPU's console serial port, set up IP addresses, work with the Fujitsu AXEL-X2 reference software for basic switch management, and finally how to optionally configure routing or bridging between the two Ethernet ports. The DA112 CPU defaults to having no routing or bridging take place between the two ports, leaving the 10G and 1G Ethernet segments completely isolated from each other. The DA116 and DA152 have a dedicated link between the 10GbE and 1GbE switches. All switches support RSTP to break loops in the network.

NOTE: This document describes features available with the BSP version 9 software. If you are running a previous version then it is advisable to upgrade. Previous document versions may be available which discuss previous feature sets if you prefer to stick with an earlier software release.

1.6 Document References

- VadaTech UTC001 Hardware Reference Manual
- VadaTech UTC002 Hardware Reference Manual
- VadaTech VT851 Hardware Reference Manual
- VadaTech VT852 Hardware Reference Manual
- VadaTech VT853 Hardware Reference Manual
- AXEL-X2 MB86C69RBC Reference Software User Guide
- Ethernet Bridge + netfilter Howto (http://tldp.org/HOWTO/Ethernet-Bridge-netfilter-HOWTO.html)

1.7 Acronyms Used in this Document

Acronym	Description
1GbE	Gigabit Ethernet
10GbE	10 Gigabit Ethernet (XAUI)
AMC	Advanced Mezzanine Card
BSP	Board Support Package
CLI	Command-Line Interface
MCH	MicroTCA Carrier Hub
SFP	Small Form Factor Pluggable
SFP+	Small Form Factor Pluggable Plus
(R)STP	(Rapid) Spanning Tree Protocol
XAUI	10 Gigabit Attachment Unit Interface

Table 7: Acronyms

2 Accessing the 10GbE Switch CPU's Serial Console

2.1 Getting to the Serial Console on UTC00x/VT85x

The MCH has a single serial console port on the front panel which is used to communicate with the on-board VT002 IPMI controller module. From the VT002 console it is possible to run a terminal program to communicate with the separate switch management CPU that is located on the DA112/DA116/DA152 daughter card. Refer to the hardware reference manual for your MCH or 1U chassis for information on port pin-out, etc.

2.1.1 Setting up UTC00x/VT85x UART Routing

The VT002 CPU has several serial ports, and furthermore the one that connects to the DA112/DA116/DA152 (/dev/ttyS2) is multiplexed between two different targets (refer to the UTC001/UTC001/VT851/VT852/VT853 logical block diagram in the corresponding hardware reference manual). Because of this, the multiplexer must be selected properly before communication with the DA112/DA116/DA152 can succeed. This is done by entering the following command on the VT002 console:

```
vt set rs232sel=0
```

2.1.2 Setting up the VT002 Terminal Type and Geometry

The VT002 terminal must be configured properly before the terminal program used to connect to the DA112/DA116/DA152 will display as expected. To do this, determine the width (*WW*) and height (*HH*) of the terminal window that you are using to connect to the VT002 and then enter them as part of the following command:

```
export TERM=vt100; stty rows HH cols WW
```

2.1.3 Running Minicom on the VT002

The final step in connecting to the DA112/DA116/DA152 from the VT002 console is to run the minicom application on the VT002 as follows:

```
minicom –w –o ttyS2
```

WARNING: Make sure to pass the '`-o`' option to minicom otherwise there could be side-effects to the modem initialization that minicom tries to do when this option is not passed (such as stopping the DA112/DA116/DA152 boot loader's auto-boot sequence, etc).

2.1.4 Running Minicom on the VT002 w/ Forced Reboot

In order to gain access to the boot loader on the DA112/DA116/DA152 either the **reboot** command can be issued from the DA112/DA116/DA152 Linux console, or the following command sequence can be issued at the VT002 console:

```
vt set rstpayt=1; vt set rstpayt=0; minicom –w –o ttyS2
```

However, when instructed to perform a power-cycle (i.e. during software upgrade) please do so by removing power from the MCH and then re-applying it instead of simply doing a reset as some configuration data will only be loaded on a cold-boot.

NOTE: If you were originally connecting to the VT002 via a minicom application on your host PC then you may need to re-map the meta/command key in either your host-side minicom application or the VT002-side minicom application to have full control over both minicom instances. Otherwise the outermost (host PC) instance will capture the meta-commands and they will not be seen by the VT002 instance. See the 'Command key' option under the minicom 'Screen and keyboard' configuration menu. To avoid this issue we recommend that you use a different terminal program on the PC such as 'TeraTerm'.

2.2 Getting to the Serial Console on AMC229

The AMC229 has a port on the front panel labeled 'SERIAL' to which a VadaTech P/N CBLDB9MUSB1 may be connected. The DB9 end can be connected to a PC COM port with settings of 115200-8-N-1-NOFLOW.

3 Determining the Board Support Software (BSP) version

The BSP software version can be determined by running this command:

```
bsp-version
```

NOTE: Prior to BSP version 7 this command is not available and the BSP version was generally tracked by the '-vtN' suffix of the Linux kernel and/or the version number of the U-Boot bootloader. If you have difficulty determining your BSP version please contact VadaTech for further instruction.

4 Setting IP Addresses

The management CPU ships from VadaTech with the following IP address configuration:

eth0: **192.168.41.222** (on the 10GbE segment)
eth1: **192.168.40.222** (on the 1GbE segment)

These IP addresses can be re-assigned by editing the **`/etc/rc.d/rc.conf`** file. After any of the files in **`/etc/rc.d`** are changed, the '**`reboot`**' command should be issued so that the CPU will restart and reload the new configuration.

When two or more boards are installed into the same chassis or their networks are otherwise connected it is important to re-assign the IP addresses to avoid conflicts. It is recommended to do this with only one board attached to the network/backplane at a time.

Once IP addressing has been established it is possible to use SSH as an alternative to using the serial console.

NOTE: On AMC229 'eth1' doesn't connect to anything and can be disabled if desired by commenting out the appropriate lines in /etc/rc.d/rc.conf.

5 AXCLI

The management CPU comes pre-loaded with the Fujitsu AXEL-X2 reference software (called 'axcli'). This software provides a command-line interface to control the 10GbE switch. The mechanism for control is via in-band management frames sent via eth0 (or br0 if eth0 and eth1 are bridged) to a special MAC address. The 10GbE switch recognizes this MAC address and routes the frames to the switch's internal microcode engine.

You can access the CLI as follows:

```
axcli
```

For a complete description of how to use this CLI please refer to the AXEL-X MB86C69RBC Reference Software User Guide.

NOTE: The **axdrvd** service used by **axcli** to communicate to the switch chip requires a special configuration if the management CPU's Ethernet ports are setup for Ethernet Bridging. Please refer to the sections that follow for more information.

Starting with BSP version 7 the **axcli** configuration files are stored in the **/etc/axelx** directory. Prior to this BSP they were stored in **/var/axelx**. This change of location was necessitated because the contents of the **/var** directory tree are re-initialized on every reboot. By storing the files in **/etc/axelx** the settings will remain persistent across reboots. This directory also contains an extra read-only copy of each configuration file which can be copied over the active copy in order to restore the factory defaults.

6 Enabling IP Routing

NOTE: This section is not applicable to AMC229 since its 'eth1' port has no connection.

The switch management CPU can be configured to do IP Routing between its two Ethernet ports. By default this feature is turned off. To turn on routing, execute the following command:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

NOTE: To make this setting persistent, simply add the command to the end of the **`/etc/rc.d/rc.local`** file.

IP packets received by the management CPU will be routed to their destination based on the Linux kernel's standard routing rules. You will need to setup routes on the other nodes of the network to specify the appropriate IP address as a gateway router (using the IP address of the Ethernet interface that faces the node on the same segment). You may also wish to configure a default route on the management CPU to provide a pathway for packets that are travelling beyond the two physically connected networks local to the board.

With a routing configuration, broadcast traffic is blocked between the two segments but directed unicast traffic is routed between them. Therefore services such as DCHP, ARP, etc will not cross between the 10G and 1G segments. If your system configuration requires the sharing of this type of broadcast-dependent service between the 10G and 1G segments refer to the Ethernet Bridging section that follows.

7 Enabling Ethernet Bridging

NOTE: This section is only applicable to DA112. Other boards do not require Ethernet Bridging of eth0 and eth1.

The DA112 CPU may be configured to bridge its two Ethernet ports together at Layer 2 if it is desired to have the 10Gb and 1Gb segments appear at one contiguous LAN. This effectively turns the CPU into a two port Ethernet switch (with optional STP support). Furthermore, packet filtering via the 'iptables' command is also available for advanced configurations such as firewalling, etc. Once the Ethernet ports are bridged, 'eth0' and 'eth1' belong to the bridge named 'br0'.

To enable bridging of the two Ethernet ports together look in the /etc/rc.d/init.d/network script and uncomment the bridging commands as instructed in the script. Then also look in the /etc/rc.d/init.d/axdrvd script and change the driver configuration to the secondary configuration as described in the script. This causes axdrvd to use 'br0' as the switch management port instead of 'eth0'. Reboot when the changes are complete to activate the bridge.

The 10G and 1G Ethernet segments should now be bridged together. All frames seen on one interface will be forwarded directly to the other interface as would happen in a typical Ethernet switch. Be aware that the DA112 has only a single IP address on the newly bridged segment whereas before it had one IP address on each isolated/routed segment.

If Ethernet Bridging is enabled on the CPU, it acts like a 3-port switch connecting 'eth0', 'eth1', and the local node and performs STP operations on those ports by default. Therefore there is a delay in the ports forwarding frames once the bridge is brought up. The 'axdrvd' driver will not function properly until the STP learning process has completed. If it is not desirable to wait for the ports to enter forwarding mode before starting the switch management daemon, it is possible to disable the STP implementation in the bridge using the command:

```
brctl stp br0 off
```

To undo the bridging, simply reverse the changes made before and reboot. Dynamic bridging is possible, but this requires advanced coordination of the various daemons and is therefore left up to the application-specific scripting done by the customer.

NOTE: When the Ethernet ports are bridged, one 10GbE switch management frame will leak out to the 1G segment when the 'axdrvd' service is started. This should be harmlessly dropped by the other nodes on the 1G segment as an unrecognized MAC address. But it may be picked up by a network sniffer as an unusual frame and should not be a cause for alarm. After this first frame is responded to by the 10GbE switch, the CPU's Ethernet Bridge

functionality will learn the switch's MAC address and stop flooding subsequent management packets to the 1G segment.

Please refer to the Linux Bridging HOWTO available on the web for detailed background on bridging and bridging operations.

8 Determining Link Status

The link status can be confirmed via the 'axcli', however, this can be a tedious process. Therefore a convenient shell command named **axel_linkstat** is also included to simplify the reporting of link status. An example (taken from a DA116 board) is shown below:

```
~ # axel_linkstat
(AMC 01) Port 15: NO LINK
(AMC 02) Port  7: NO LINK
(AMC 03) Port  0: NO LINK
(AMC 04) Port  8: LINK
(AMC 05) Port 16: NO LINK
(AMC 06) Port 13: NO LINK
(AMC 07) Port 19: NO LINK
(AMC 08) Port 11: NO LINK
(AMC 09) Port  3: LINK
(AMC 10) Port  4: NO LINK
(AMC 11) Port 12: NO LINK
(AMC 12) Port 17: NO LINK
(UPDATE) Port 18: NO LINK
(SFP+ 0) Port 14: NO LINK
(SFP+ 1) Port 10: NO LINK
(SWTOSW) Port  1: LINK
(MANAGE) Port  9: LINK
```

9 Changing Port Speeds

SFP+ ports on these boards are auto-detected at boot time and the port speeds set accordingly. But if SFP+ modules are switched out without a reboot then it would be necessary to change the port speed manually. Also backplane ports do not automatically change speed so these would also need to be configured manually if the default is not desired.

The following scripts are available for changing port speeds/configurations:

`axel_1g_port X`	(converts physical port X to 1000Base-X)
`axel_xaui_port X`	(converts physical port X to XAUI)
`axel_10g_port X`	(converts physical port X to 10GBase-KR)

To make this port configuration persistent across reboots simply put the needed commands at the end of the **`/etc/rc.d/rc.local`** script.

Please contact VadaTech for support if you need special e-keying for backplane ports.

NOTE: The switch chip does not support 1000Base-X auto-negotiation. Please disable this auto-negotiation on the remote side prior to using a 1GbE port. Otherwise the switch will shutdown the port.

10 RSTP

An RSTP implementation is active on the 10GbE switch by default. Most of the switch ports start out powered-down via the switch's EEPROM until the RSTP service can start functioning under Linux. This is to ensure that no loop can be formed in the network. Once the RSTP service is started, the ports are powered up and the RSTP learning process takes place which finally switches the non-looped ports into forwarding mode.

On the AMC229 a special path is left powered up so that **`eth0`** can connect to SFP+ 0 on the front panel at 1000Base-X speed directly out of power-on. This enables U-Boot Ethernet connectivity for software upgrade / maintenance. The SFP+ speed detection is not enabled while running under U-Boot, therefore you cannot connect to the board at 10GbE speed when in the boot loader.

The RSTP implementation consists of the following components:

1) The **`axdrvd`** daemon which provides communication to the switch chip
2) The **`axnimd`** daemon which provides 'virtual' TAP/TUN switch ports
3) The **`br1`** TAP/TUN bridge which contains all of the 'virtual' switch ports
4) The **`rstpd`** daemon which implements the protocol and state machine

The structure that is created for managing the switch looks like this:

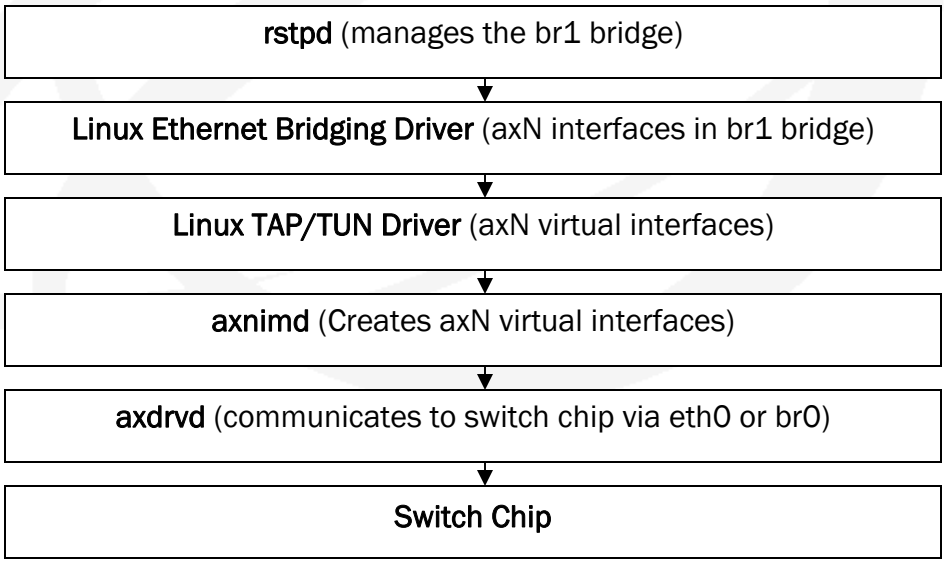

Figure 6: RSTP implementation

NOTE: The rstpd daemon takes over from the STP implementation in the kernel. Each time a port is overridden the kernel implementation issues a log message about the port being disabled. These messages should simply be ignored.

An **rstpstat** script is provided which displays the RSTP bridge information as well as each physical port's state along with the logical port name for the detected board type. An example from the AMC229 board is shown below:

```
/ # rstpstat
Bridge:          br1                   State:enabled
BridgeId:        8000-02133a002906     Bridge Proirity: 32768 (0x8000)
Designated Root: 8000-02133a002906
Root Port:       none
Time Since Topology Change: 4138
Max Age:         20   Bridge Max Age:       20
Hello Time:       2   Bridge Hello Time:    2
Forward Delay:   15   Bridge Forward Delay: 15
Hold Time:        3
(SFP+ 0) *    ax8 8009 Fwd  8000-02133a002906 8000-02133a002906 8009 D
(SFP+ 1) *    ax4 8005 Blk  8000-02133a002906 8000-02133a002906 8001 B
(SFP+ 2) *    ax7 8008 Fwd  8000-02133a002906 8000-02133a002906 8008 D
(SFP+ 3) *    ax11 800b Blk  8000-02133a002906 8000-02133a002906 8009 B
(SFP+ 4) *    ax12 800c Fwd  8000-02133a002906 8000-02133a002906 800c D
(SFP+ 5) *    ax0 8001 Fwd  8000-02133a002906 8000-02133a002906 8001 D
(SFP+ 6) *    ax3 8004 Fwd  8000-02133a002906 8000-02133a002906 8004 D
(SFP+ 7) *    ax15 800f Fwd  8000-02133a002906 8000-02133a002906 800f D
(BASE 0) *    ax13 800d Fwd  8000-02133a002906 8000-02133a002906 800d D
(BASE 1) *    ax16 8010 Fwd  8000-02133a002906 8000-02133a002906 8010 D
(F4-7  ) *    ax14 800e Fwd  8000-02133a002906 8000-02133a002906 800e D
(F8-11 ) *    ax18 8012 Fwd  8000-02133a002906 8000-02133a002906 8012 D
(F17-20) *    ax17 8011 Fwd  8000-02133a002906 8000-02133a002906 8011 D
(MANAGE) [ax9 not part of RSTP bridge]
```

If RSTP is not desired, the **axnimd**, **rstpd**, and **axbrctl** startup script invocations can be commented out of the **/etc/rc.d/init.d/network** script.

The following diagram shows some of the possible loops that can be formed in a uTCA backplane network; demonstrating the need for RSTP to be enabled on all of the various switches or the use of manual port disabling in order to avoid a broadcast storm.

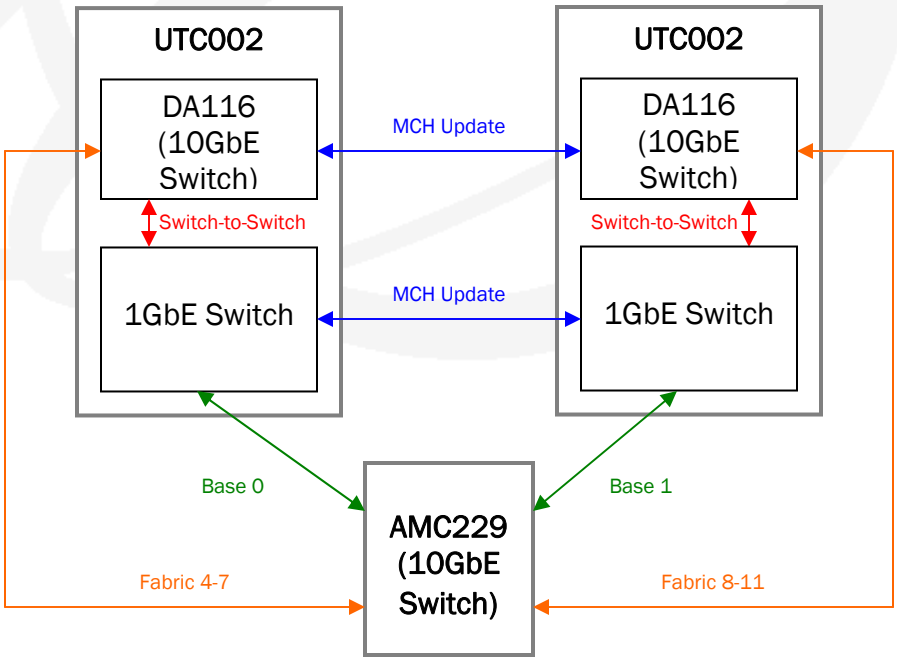

Figure 7: uTCA backplane network loop example

11 Additional Tools

The following additional tools are available for use when doing custom scripting, etc.

NOTE: The detailed chip specification for the AXEL-X2 switch chip is only available from Fujitsu under NDA. VadaTech cannot provide this document nor the source code for the AXEL-X2 reference software. Customers will have to coordinate an NDA with Fujitsu directly to obtain these materials if they want to do direct access to the switch chip.

11.1 axel_i2c

The **axel_i2c** tool provides raw access to the switch chip's registers, a pathway for reprogramming the switch's EEPROM (please use only VadaTech provided images), SFP+ speed detection/EEPROM readout, and SFP+ GPIO line state dumping (AMC229 only).

NOTE: Using this tool to write to the switch's registers is NOT recommended as it can conflict with the switch's on-chip microcode engine and cause unpredictable results. See the **axel_reg** command below for a safer alternative.

```
Usage: axel_i2c <cmd> <args>

CMD:
        read <regaddr> [<count>]          -  Read 'count' (or 1) register(s)
                                             starting at 'addr'
        write <regaddr> <value>           -  Write 'value' to register at 'addr'
        eeprom_read <bin/ascii> <file>    -  Read the AXEL-X2 EEPROM contents
                                             and write to bin/ascii file
        eeprom_write <bin/ascii> <file>   -  Write the AXEL-X2 EEPROM contents
                                             from bin/ascii file
        sfp_detect <0-7>                  -  Detect SFP speed
        sfp_read <0-7> <bin/ascii> <file> -  Read the SFP EEPROM contents
                                             and write to bin/ascii file
        sfp_gpio                          -  Show SFP GPIO line states
```

11.2 axel_reg

The axel_reg command provides switch chip register access and restart capability by using the in-band communication mechanism. This is the recommended way of reading/writing registers since it synchronizes with the switch chip's microcode engine correctly.

```
usage: axel_reg <get[v]|set[v]> chip <reg_offset> [<value>]
       axel_reg <get[v]|set[v]> port <port_idx> <reg_offset> [<value>]
       axel_reg <get[v]|set[v]> pcs <port_idx> <dev> <reg_offset> [<value>]
       axel_reg restart

NOTES:
  The [v] options (getv, setv) have verbose output.
  The restart option is always verbose and it is normal
  for it to complain about a timeout.
```

11.3 axel_reg_get_all_ports

This script is a wrapper for the axel_reg command and provides cross-sectional view of a port register across all ports based on the detected board type.

Example from the AMC229 board:

```
/usr/bin # axel_reg_get_all_ports 0x0000
(SFP+ 0) Port 08 [0x0000]: 0x59030001
(SFP+ 1) Port 04 [0x0000]: 0x59010001
(SFP+ 2) Port 07 [0x0000]: 0x59030001
(SFP+ 3) Port 11 [0x0000]: 0x59010001
(SFP+ 4) Port 12 [0x0000]: 0x59030001
(SFP+ 5) Port 00 [0x0000]: 0x59030001
(SFP+ 6) Port 03 [0x0000]: 0x59030001
(SFP+ 7) Port 15 [0x0000]: 0x59030001
(BASE 0) Port 13 [0x0000]: 0x59030001
(BASE 1) Port 16 [0x0000]: 0x59030001
(F4-7  ) Port 14 [0x0000]: 0x51030001
(F8-11 ) Port 18 [0x0000]: 0x51030001
(F17-20) Port 17 [0x0000]: 0x51030001
(MANAGE) Port 09 [0x0000]: 0x59030001
```

11.4 axel_reg_get_all_pcs

This script is a wrapper for the axel_reg command and provides cross-sectional view of a PCS register across all ports based on the detected board type.

Example from the AMC229 board:

```
/usr/bin # axel_reg_get_all_pcs 3 0x0000
(SFP+ 0) PCS 08 [3.0000]: 0x00002040
(SFP+ 1) PCS 04 [3.0000]: 0x00002040
(SFP+ 2) PCS 07 [3.0000]: 0x00002040
(SFP+ 3) PCS 11 [3.0000]: 0x00002040
(SFP+ 4) PCS 12 [3.0000]: 0x00002040
(SFP+ 5) PCS 00 [3.0000]: 0x00002040
(SFP+ 6) PCS 03 [3.0000]: 0x00002040
(SFP+ 7) PCS 15 [3.0000]: 0x00002040
(BASE 0) PCS 13 [3.0000]: 0x00002040
(BASE 1) PCS 16 [3.0000]: 0x00002040
(F4-7  ) PCS 14 [3.0000]: 0x00002040
(F8-11 ) PCS 18 [3.0000]: 0x00002040
(F17-20) PCS 17 [3.0000]: 0x00002040
(MANAGE) PCS 09 [3.0000]: 0x00002040
```

11.5 GPIO Resets

The BSP's kernel code exports a subsystem which can be used for controlling the on-board GPIO reset lines. These lines can be used to reset various parts of the board. The only one that should be of use to the customer is the **switch_reset** which can be used to reset the switch chip and cause it to reload its EEPROM.

11.5.1 Switch Reset

```
echo 1 > /sys/mgmt_gpio/switch_reset      (assert switch reset)
echo 0 > /sys/mgmt_gpio/switch_reset      (de-assert switch reset)
```

The switch reset affects the 10GbE switch chip; therefore it is recommended that all switch management daemons be shut down prior to executing this reset.

NOTE: Simply 'reboot'ing the management CPU does NOT reset the switch, so be sure to issue a switch reset prior to rebooting if you are trying to restart the whole board cleanly. A RSTPAYT reset from the MCH baseboard via the VT002 management controller does reset both the CPU and the switch chip.

11.5.2 SPI Reset

```
echo 1 > /sys/mgmt_gpio/spi_reset    (assert SPI reset)
echo 0 > /sys/mgmt_gpio/spi_reset    (de-assert SPI reset)
```

The SPI reset affects the SPI to I2C bus bridge chip. Use of this reset is not recommended without instruction from VadaTech.

11.5.3 I2C Reset

```
echo 1 > /sys/mgmt_gpio/i2c_reset    (assert I2C reset)
echo 0 > /sys/mgmt_gpio/i2c_reset    (de-assert I2C reset)
```

The I2C reset affects the I2C SFP+ hub chip as well as the SFP+ GPIO chip (AMC229 only). Use of this reset is not recommended without instruction from VadaTech.

12 Software Upgrade

An easy-to-use software upgrade mechanism is provided when field upgrades are issued. The upgrade steps involve transferring the upgrade package (tgz file) to the board's /tmp directory using scp, exploding it (**`tar xvzf <filename>`**), and then running the **`upgrade`** script that results.

The upgrade script attempts to identify what board it is running on and to do the appropriate actions including upgrading the firmware components and rewriting the switch EEPROM. If the upgrade script is running on some earlier BSP versions it will not be able to automatically determine the difference between a DA112 and DA116 board and you will need to provide this information. First run the upgrade script with no arguments and then if the script complains that it can't automatically determine your board type then you may need to provide one of the following options to help it out:

`--da112`	(If you have a DA112 (UTC001) board that the script can't detect)
`--da116`	(If you have a DA116 (UTC002) board that the script can't detect)

The upgrade script attempts to minimize the number of steps needed by determining if any of the partition images hasn't changed and skipping them (unless you specify a **`--force`** argument).

For upgrade safety, the Linux Kernel and Root Filesystem partitions are redundant. The active partitions are not erased or written during the upgrade; instead inactive partitions are erased/written and then activated once all of the partitions are in a known state.

NOTE: Any customizations that were previously done to the filesystem will be lost during the upgrade and other things like SSH keys will be regenerated. Therefore it is advisable to save off any changes that you want to preserve and reapply them after the upgrade. Keep in mind however that scripts may have changed substantially between BSP versions, so please review the changes prior to reapplying them to be sure they are still appropriate.

After the upgrade is completed and the power is cycled, it is normal for the board to sit for a minute or two during the Linux boot; apparently doing nothing. The operating system is initializing the new filesystem during this time. Please be patient and wait for the initialization to complete.