DSS NETWORKS

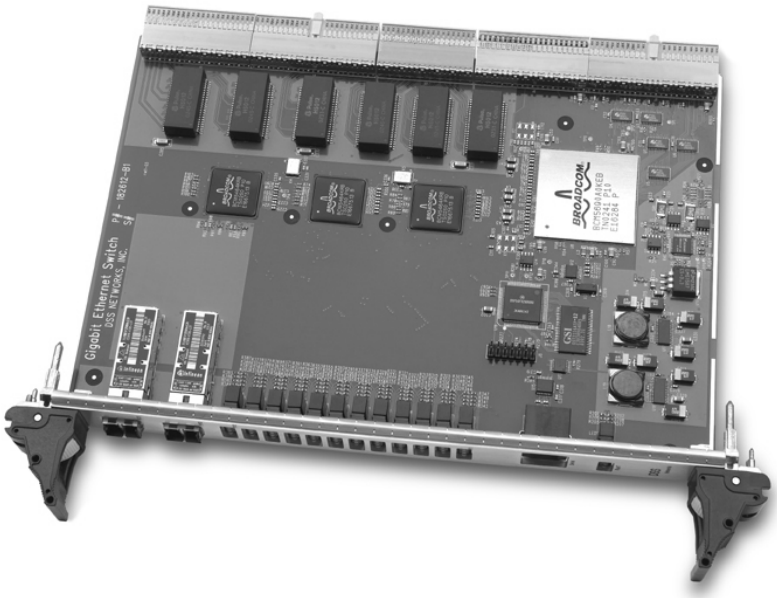
Metro-Switch Gigabit Backplane Switch Fabric

Board and Firmware Users Manual

(Includes models 8260, 8261 and 8261-RIO)

Firmware Version 1.14g, Document p/n 131902

May 2007



1. INTRODUCTION

The Metro-Switch family of embedded Gigabit Ethernet switches are a high-performance, flexible and cost-effective solution for adding scalable Gigabit Ethernet switching capabilities for use in standalone or rackmount embedded systems applications.

The Models 8260 and 8261 support 12-ports of 10/100/1000 Base T over copper or 10-ports copper and 2 ports of 1000 Base X over optical fiber. The Model 8260 is a L2+ or L3 switch utilizing lower cost switch L2+ and L3 switch devices from Broadcom. The Model 8261 uses higher end multi-layer (L2 – L7) switch devices also in the Broadcom StrataXGS family. Both models provide an extensive list of features including an onboard management processor with extensible value-added firmware making it ideal for use in many embedded systems applications.

1.1 CAPABILITIES OVERVIEW

The Metro-Switch models feature 12 ports of 10/100/1000 Base T Gigabit Ethernet over Copper with two 1000 Base SX/LX fiber uplinks. It is PICMG 2.16 6U fabric card compliant and compatible with both standard CompactPCI® and PICMG 2.16 backplanes. All 12-ports may be routed to slots on the Compact PCI backplane or externally via rear I/O. A system management interface is also supported via the PICMG 2.9 IPMI interface. It optionally supports two 1000 base SX/LX gigabit fiber ports with standard SFF LC connectors via the front panel. It has an onboard RISC/DSP processor for local management and can be operated as a standalone or fully managed switch. LEDS are provided for each port showing link status, transmit and receive and link quality. All LEDS are multifunction and can be used for additional functions including cable testing and energy detection. It is also PICMG 2.1 R2.0 hot-swap compliant providing support for the hardware connection layer.

The Metro-Switch models use the latest advanced high-performance, full-featured and highly integrated 12-port Broadcom StrataSwitch BCM56XX L2+, L3 or multilayer switches and BCM5464SR quad-port transceivers and are fully 802.3 compliant. They provide a fully non-blocking 24Gb/32 million frames per second aggregate switching fabric. The switching function supports an extended list of features including layer 3 switching, link aggregation, 802.1Q VLANs, 802.1D spanning tree and priority-based 802.1D/802.1p CoS/traffic class expediting and dynamic multicast filtering. The switch can be configured in a fully redundant, non-blocking network that prevents single points of failure from congesting network traffic. It provides advanced cell and packet based “head of line” blocking prevention techniques, has 1MB of onboard memory for packet buffering, and supports a 10-gig uplink interface. Extended ethernet frame sizes to 9KB are supported. Additional advanced features including rules-based layer 2-7 packet classification/filtering on 128 multiple data flows, port trunking and port mirroring are provided for advanced networking and flow techniques. Network management support

includes fully configurable routing tables and RMON, SNMP, Ethernet and extended MIB(s). A 32-bit, 66 MHZ PCI interface is also provided to support system management via the PCI bus and the switch can additionally route packets to/from the PCI interface as a “virtual port” function. This would allow for example, SNMP or other management packets to be routed to an external processing component in a distributed network management scheme.

1.2 USE AND TARGET APPLICATIONS

The Metro-Switch 12-port switches are equipped ready-to-use out of the box with no additional software or drivers required. In this most basic operational mode it is used as a ***standalone switch*** installed in a rackmount chassis. An additional mode allows it to be controlled in a ***simple managed mode*** using a serial port ***Command Line Interface*** (CLI). The CLI allows more sophisticated configuration and control management while still operating as a standalone switch. The most functional mode allows it to be controlled in a ***fully managed switch*** mode from an external processor communicating with the Metro-Switch over the PCI bus via the JN1 CPCI connector.

The Metro-Switch models are targeted for OEMs and Systems Integrators for use in Data and Telecommunications products including switches, multiplexers, edge routers, media gateways and video broadcasting equipment. It is well suited for support of embedded broadband applications including Internet voice, digital video, IP security, network monitoring, military applications and test equipment.

1.3 KIT CONTENTS

- Metro-Switch 12-port board with front panel mounting and ejector handles
- Serial-port RJ11 male to DB9 female cable
- Telnet adapter module (ordered under a separate P/N or customer supplied)
- User Manual
- OEM Developer Kit CD

1.4 REFERENCES

Please also see the following documents on our website at www.dssnetworks.com and also included in the OEM developers kit CD:

Datasheets – please see product datasheets and other updated product information on OEM developer CD and on website.

Release Notes -- where updated information is provided on new features, compatibility, performance benchmarks, platform information and corrected problems.

1.5 COMPATIBILITY

The Metro-Switch is fully compliant with the following standards:

- IEEE 802.3-2002 (all sections applicable to 1000 Base T, 1000 Base SX, 1000 Base LX)
- IEEE 802.1D and IEEE 802.1Q as applicable for VLAN and priority queuing support
- PCI 2.2 bus compliant, 32-bit 33/66 MHZ
- PICMG 2.0 R3.0 Compact PCI compliant (6U form factor)
- PICMG specifications: 2.16 R1.0 Packet Switching Backplane (PSB) complaint fabric board
- PICMG 2.9 IPMI support
- PICMG 2.1 R2.0 hot-swap (basic hot-swap hardware connection layer)
- Telnet remote terminal access standard

2. MODEL / PART NUMBERS

This user manual covers all models of our Metro-Switch product line including:

Model 8260

Part Number	182602
Model	8260
Description	12-port Compact PCI PICMG 2.16 compliant switch
Copper ports	12 (10/100/1000 Base T)
Fiber ports	2 (1000 Base SX 850nm multimode LC fiber)
Switch type	L2+ and L3

Part Number	182606
Model	8260
Description	12-port Compact PCI PICMG 2.16 compliant switch
Copper ports	12 (10/100/1000 Base T)
Fiber ports	2 (1000 Base LX 1310nm singlemode LC fiber)
Switch type	L2+ and L3

Part Number	182604
Model	8260
Description	12-port Compact PCI PICMG 2.16 compliant switch
# Copper ports	12 (10/100/1000 Base T)
# Fiber ports	0 (without fiber uplinks)
Switch type	L2+ and L3

Model 8261

Part Number	182612
Model	8261
Description	12-port Compact PCI PICMG 2.16 compliant switch
Copper ports	12 (10/100/1000 Base T)
Fiber ports	2 (1000 Base SX 850nm multimode LC fiber)
Switch type	Multilayer (L2 – L7)

Part Number	182616
Model	8261
Description	12-port Compact PCI PICMG 2.16 compliant switch
Copper ports	12 (10/100/1000 Base T)
Fiber ports	2 (1000 Base LX 1310nm singlemode LC fiber)
Switch type	Multilayer (L2 – L7)

Part Number	182614
Model	8261
Description	12-port Compact PCI PICMG 2.16 compliant switch
# Copper ports	12 (10/100/1000 Base T)
# Fiber ports	0 (without fiber uplinks)
Switch type	Multilayer (L2 – L7)

Note: Please note that copper ports 10 and 11 are not active when fiber ports are used.

3. FEATURES

The Metro-Switch adapter offers the following key features:

- 12-port full-duplex Gigabit Ethernet interfaces routed to 2.16 backplane or rear I/O
- Optional support for 2 multimode or singlemode fiber links (850nm MM or 1310nm SM LC)
- Sustained aggregate throughput of 24 Gbps (3 GB)
- Frame processing rate of up to 32 million frames per
- Onboard RISC management processor and EEPROM configuration
- 4 multi-function LEDS per port (TX, RX, LINK, Signal Quality)
- 32-bit, 66 MHZ PCI interface
- Onboard RS232 (RJ-11 jack) for serial port console support
- Remote network access via Telnet
- Basic hot-swap support for hardware connection layer
- Installs in CompactPCI PICMG 2.16 compliant system chassis
- Optionally installs in standard Compact PCI PICMG 2.0 R3.0 chassis
- Available with 12-port rear I/O transition module
- Ideal solution for CompactPCI rackmount embedded systems
- Utilizes single 5V power from JN1/JP1 PCI connector
- Provides power regulators onboard for power distribution (3.3V, 2.5V, 1.25V)
- Complies with all PCI revision 2.2 mechanical and electrical requirements
- Fully IEEE 802.3z, IEEE 802.3ab, 802.3u and IEEE 1386 compliant
- Compatible with all 10/100/1000BaseT hubs, switches and routers
- Operates as standalone Layer 2 switch or fully managed L3/multiplayer switch
- Jumbo frame support for up to 9K
- 802.3x full duplex flow control

4. SYSTEM REQUIREMENTS

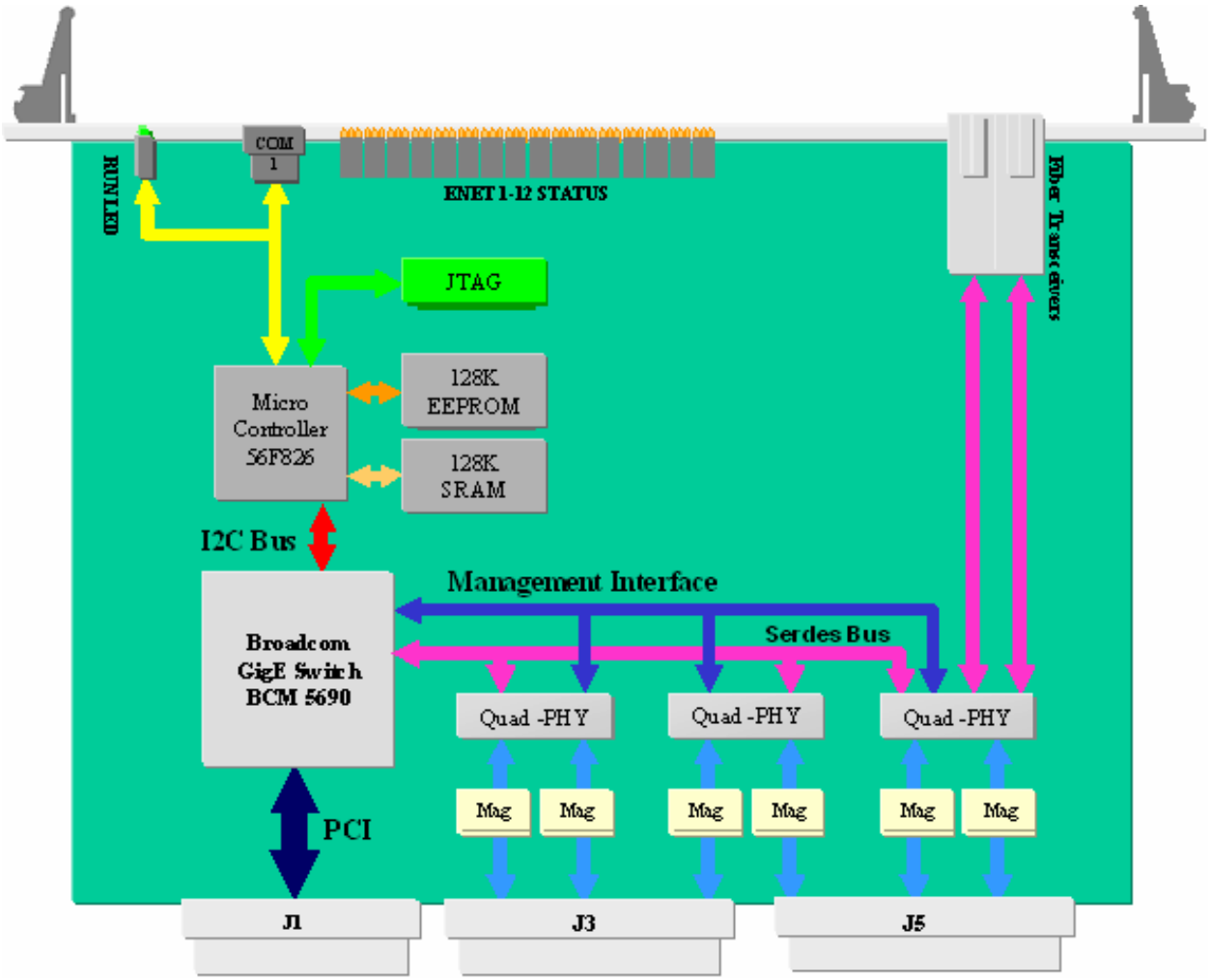
Compact PCI PSB Platform:

- Compact PCI compliant (PICMG 2.16, PICMG 2.0 R3.0)
- Size 6U chassis and slots, power supply and fan(s)
- Packet Switching Backplane Compliant to PICMG 2.16 R1.0
- At least one fabric slot required
- Metro-Switch installs into fabric slot in PICMG 2.16 PSB backplane

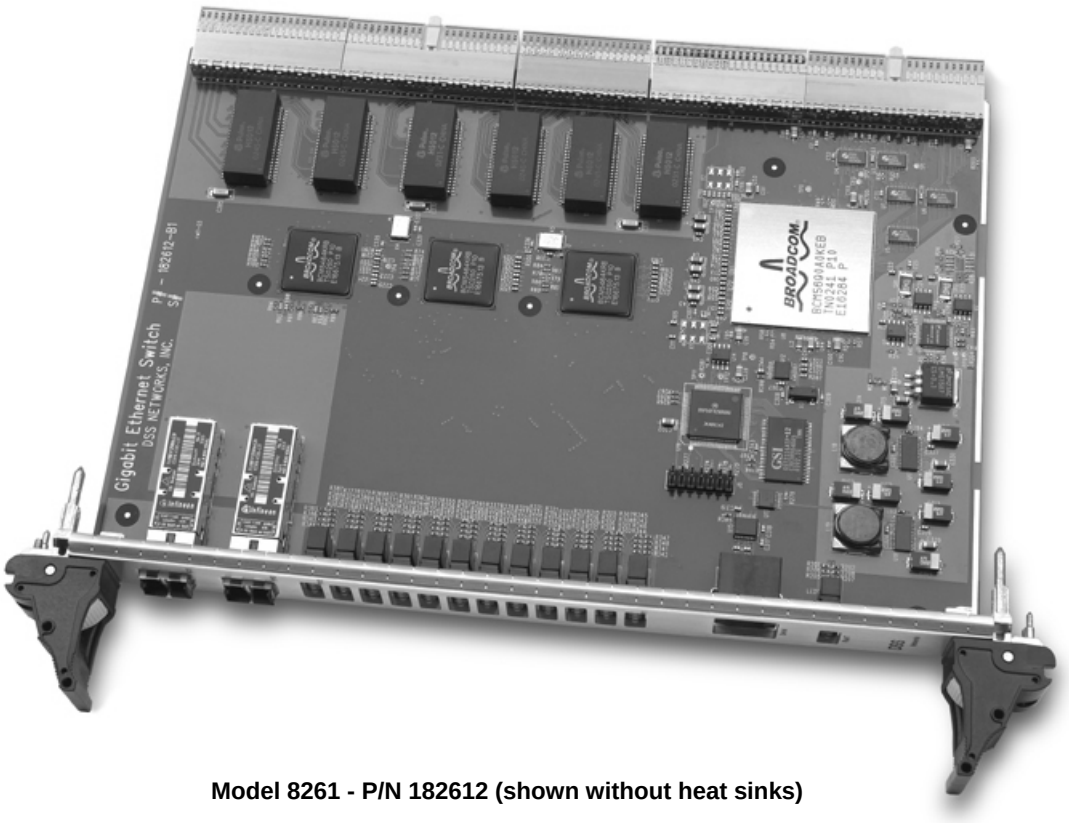
Note: The Metro-Switch may also be installed into peripheral slot on standard Compact PCI chassis with H.110 compatible backplane for connections via rear I/O only, however all pins on CPCI connectors “J3” and “J5” must be passive and comply to PICMG 2.16 signal definitions and routed to rear I/O.

5. BOARD AND CONNECTOR INFORMATION

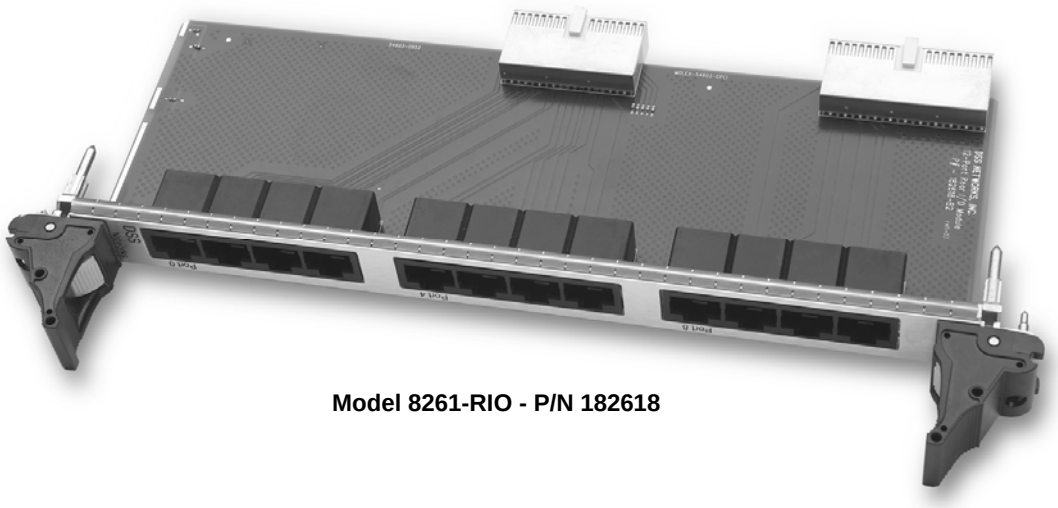
5.1 COMPONENT DIAGRAM



5.2 BOARD PHOTOS



Model 8261 - P/N 182612 (shown without heat sinks)



Model 8261-RIO - P/N 182618

5.3 BOARD LED INDICATORS

There are 12 LED arrays marked “P0” thru “P11” on the front panel. Each port array has 4 green LEDS and their meaning is described in the following table:

Port LED Arrays (P0 – P11) Function Table (from left to right)

Gigabit Switch Model	LED # 1 Green	LED # 2 Green	LED #3 Yellow	LED #4 Yellow
8260/8261	Link	TX	RX	Quality*

Note: The Quality LED is only available for ports in 1000-Base-T copper mode.

There is also a single LED Array containing a row of 3 LEDS (2 Yellow, 1 Green) described in the following table:

Processor LED (LED1) Function Table (left to right)

Gigabit Switch Model	LED # 1 YEL	LED # 2 YEL	LED #3 Green
8260/8261	Warn	LinkSt	RUN

Note: After power on and during normal operation, the green “RUN” LED should always blink at a slow steady heartbeat pace of about once per every 2 seconds.

5.4 CPCI CONNECTOR PIN/SIGNAL DEFINITIONS

The MetroSwitch models use the J1, J3 and J5 Compact PCI connectors as shown in the following diagrams. Connectors J2 and J4 are not used.

Board Connectors pin diagram J1

CPCI Connector J1	Signal Name	CPCI Connector J1	Signal Name
A1	EARLY_5V	B1	-12V
A2	P_TCK	B2	EARLY_5V
A3	P_INTA_N	B3	INTB#
A4	P_IPMB_PWR	B4	P_HEALTHY_N
A5	NC	B5	RESERVED
A6	P_REQ_N	B6	GND/PCI_PRSNT#
A7	P_AD30	B7	P_AD29
A8	P_AD26	B8	GND
A9	P_CBEN3	B9	P_IDSEL
A10	P_AD21	B10	GND
A11	P_AD18	B11	P_AD17
A12	NC	B12	NC
A13	NC	B13	NC
A14	NC	B14	NC
A15	EARLY_3V	B15	P_FRAME_N
A16	P_DEVSEL_N	B16	P_PCIXCAP
A17	EARLY_3V	B17	P_IPMB_SCL
A18	P_SERR_N	B18	GND
A19	EARLY_3V	B19	P_AD15
A20	P_AD12	B20	GND
A21	EARLY_3V	B21	P_AD9
A22	P_AD7	B22	GND
A23	EARLY_3V	B23	P_AD15
A24	P_AD1	B24	EARLY_5V
A25	EARLY_5V	B25	REQ64#
C1	P_TRST_N	D1	+12V
C2	P_TMS	D2	P_TDO
C3	INTC#	D3	EARLY_5V
C4	EARLY_VIO	D4	INTP
C5	P_RST_N	D5	GND
C6	EARLY_3V	D6	P_PCICLK
C7	P_AD28	D7	GND
C8	EARLY_VIO	D8	P_AD25
C9	P_AD23	D9	GND
C10	EARLY_3V	D10	P_AD20
C11	P_AD16	D11	GND
C12	NC	D12	NC
C13	NC	D13	NC
C14	NC	D14	NC
C15	P_IRDY_N	D15	P_BDSEL_N
C16	EARLY_VIO	D16	P_STOP_N

C17	P_IPMB_SDA	D17	GND
C18	EARLY_3V	D18	P_PAR
C19	P_AD14	D19	GND
C20	EARLY_VIO	D20	P_AD11
C21	P_AD8	D21	P_M66EN
C22	EARLY_3V	D22	P_AD6
C23	P_AD3	D23	EARLY_5V
C24	EARLY_VIO	D24	P_AD0
C25	P_ENUM_N	D25	EARLY_3V
E1	EARLY_5V	F1	GND
E2	P_TDI	F2	GND
E3	INTD#	F3	GND
E4	INTS	F4	GND
E5	P_GNT_N	F5	GND
E6	P_AD31	F6	GND
E7	P_AD27	F7	GND
E8	P_AD24	F8	GND
E9	P_AD22	F9	GND
E10	P_AD19	F10	GND
E11	P_CBEN2	F11	GND
E12	NC	F12	NC
E13	NC	F13	NC
E14	NC	F14	NC
E15	P_TRDY_N	F15	GND
E16	LOCK#	F16	GND
E17	P_PERR_N	F17	GND
E18	P_CBEN1	F18	GND
E19	P_AD13	F19	GND
E20	P_AD10	F20	GND
E21	P_CBEN0	F21	GND
E22	P_AD5	F22	GND
E23	P_AD2	F23	GND
E24	ACK64#	F24	GND
E25	EARLY_5V	F25	GND

Board Connectors pin diagram J3

CPCI Connector J3	Signal Name	CPCI Connector J3	Signal Name
A1	MDIB_P-1	B1	MDIB_N-1
A2	MDIA_P-1	B2	MDIA_N-1
A3	MDIB_P-2	B3	MDIB_N-2
A4	MDIA_P-2	B4	MDIA_N-2
A5	MDIB_P-3	B5	MDIB_N-3
A6	MDIA_P-3	B6	MDIA_N-3
A7	MDIB_P-4	B7	MDIB_N-4
A8	MDIA_P-4	B8	MDIA_N-4
A9	MDIB_P-5	B9	MDIB_N-5
A10	MDIA_P-5	B10	MDIA_N-5
A11	MDIB_P-6	B11	MDIB_N-6
A12	MDIA_P-6	B12	MDIA_N-6
A13	MDIB_P-7	B13	MDIB_N-7
A14	MDIA_P-7	B14	MDIA_N-7
A15	MDIB_P-8	B15	MDIB_N-8
A16	MDIA_P-8	B16	MDIA_N-8
A17	LP1-DB_P	B17	LP1-DB_N
A18	LP1-DA_P	B18	LP1-DA_N
A19	SGA4	B19	SGA3
C1	GND	D1	MDID_P-1
C2	GND	D2	MDIC_P-1
C3	GND	D3	MDID_P-2
C4	GND	D4	MDIC_P-2
C5	GND	D5	MDID_P-3
C6	GND	D6	MDIC_P-3
C7	GND	D7	MDID_P-4
C8	GND	D8	MDIC_P-4
C9	GND	D9	MDID_P-5
C10	GND	D10	MDIC_P-5
C11	GND	D11	MDID_P-6
C12	GND	D12	MDIC_P-6
C13	GND	D13	MDID_P-7
C14	GND	D14	MDIC_P-7
C15	GND	D15	MDID_P-8
C16	GND	D16	MDIC_P-8
C17	GND	D17	LP1-DD_P
C18	GND	D18	LP1-DC_P
C19	SGA2	D19	SGA1
E1	MDID_N-1	F1	GND
E2	MDIC_N-1	F2	GND
E3	MDID_N-2	F3	GND
E4	MDIC_N-2	F4	GND
E5	MDID_N-3	F5	GND
E6	MDIC_N-3	F6	GND
E7	MDID_N-4	F7	GND
E8	MDIC_N-4	F8	GND

E9	MDID_N-5	F9	GND
E10	MDIC_N-5	F10	GND
E11	MDID_N-6	F11	GND
E12	MDIC_N-6	F12	GND
E13	MDID_N-7	F13	GND
E14	MDIC_N-7	F14	GND
E15	MDID_N-8	F15	GND
E16	MDIC_N-8	F16	GND
E17	LP1-DD_N	F17	GND
E18	LP1-DC_N	F18	GND
E19	SGA0	F19	GND

Board Connectors pin diagram J5

CPCI Connector J5	Signal Name	CPCI Connector J5	Signal Name
A1	MDIB_P-9	B1	MDIB_N-9
A2	MDIA_P-9	B2	MDIA_N-9
A3	MDIB_P-10	B3	MDIB_N-10
A4	MDIA_P-10	B4	MDIA_N-10
A5	MDIB_P-11	B5	MDIB_N-11
A6	MDIA_P-11	B6	MDIA_N-11
A7	MDIB_P-12	B7	MDIB_N-12
A8	MDIA_P-12	B8	MDIA_N-12
A9	LP13-DB_P	B9	LP13-DB_N
A10	LP13-DA_P	B10	LP13-DA_N
A11	LP14-DB_P	B11	LP14-DB_N
A12	LP14-DA_P	B12	LP14-DA_N
A13	LP15-DB_P	B13	LP15-DB_N
A14	LP15-DA_P	B14	LP15-DA_N
A15	LP16-DB_P	B15	LP16-DB_N
A16	LP16-DA_P	B16	LP16-DA_N
A17	LP17-DB_P	B17	LP17-DB_N
A18	LP17-DA_P	B18	LP17-DA_N
A19	LP18-DB_P	B19	LP18-DB_N
A20	LP18-DA_P	B19	LP18-DA_N
A21	LP19-DB_P	B19	LP19-DB_N
A22	LP19-DA_P	B19	LP19-DA_N
C1	GND	D1	MDID_P-9
C2	GND	D2	MDIC_P-9
C3	GND	D3	MDID_P-10
C4	GND	D4	MDIC_P-10
C5	GND	D5	MDID_P-11
C6	GND	D6	MDIC_P-11
C7	GND	D7	MDID_P-12
C8	GND	D8	MDIC_P-12
C9	GND	D9	LP13-DD_P
C10	GND	D10	LP13-DC_P
C11	GND	D11	LP14-DD_P
C12	GND	D12	LP14-DC_P
C13	GND	D13	LP15-DD_P
C14	GND	D14	LP15-DC_P
C15	GND	D15	LP16-DD_P
C16	GND	D16	LP16-DC_P
C17	GND	D17	LP17-DD_P
C18	GND	D18	LP17-DC_P
C19	GND	D19	LP18-DD_P
C20	GND	D20	LP18-DC_P
C21	GND	D21	LP19-DD_P
C22	GND	D22	LP19-DC_P
E1	MDID_N-9	F1	GND
E2	MDIC_N-9	F2	GND

E3	MDID_N-10	F3	GND
E4	MDIC_N-10	F4	GND
E5	MDID_N-11	F5	GND
E6	MDIC_N-11	F6	GND
E7	MDID_N-12	F7	GND
E8	MDIC_N-12	F8	GND
E9	LP13-DD_N	F9	GND
E10	LP13-DC_N	F10	GND
E11	LP14-DD_N	F11	GND
E12	LP14-DC_N	F12	GND
E13	LP15-DD_N	F13	GND
E14	LP15-DC_N	F14	GND
E15	LP16-DD_N	F15	GND
E16	LP16-DC_N	F16	GND
E17	LP17-DD_N	F17	GND
E18	LP17-DC_N	F18	GND
E19	LP18-DD_N	F19	GND
E20	LP18-DC_N	F20	GND
E21	LP19-DD_N	F21	GND
E22	LP19-DC_N	F22	GND

6. POWER CONSUMPTION SPECIFICATIONS

All board power is derived from 5V power rail to CPCI “J1” connector.

Note: The 3.3V onboard power has its own power regulator taken from the 5V rail, however it can optionally be jumpered to use the 3.3V power rail from the J1 CPCI connector.

Onboard power supplies

1.2V Power (mA, A)		2.5V Power (mA, A)		3.3V Power (mA, A)		Power (W)
6.584A	7.91W	3.01A	7.53W	1.025	3.39W	18.83W

J1 CPCI Connector Power Rails

3.3V Source Current (mA, A)	5V Source (main supply) Current (mA, A)	12V power rail
Not used ***	5A (max)	Not used

***** Note: Board can be optionally jumpered at factory to use 3.3V onboard from 3.3V power rail for power distribution.**

7. HARDWARE INSTALLATION

Before attempting to install the Metro-Switch into your system, please make sure to check and verify the following:

Shut off the power to the system and chassis and any peripherals. It is important to **remove the power cable** to the system chassis.

Step 1: Assess system power requirements. If you already have other CPCI cards in your system, make sure that your system is able to provide the necessary power to support the addition of the Metro-Switch fabric card. Check your systems user manual for power specifications and limitations.

Step 2: Ground yourself. Many electronic components inside computer and on the Metro-Switch can be severely damaged by receiving a shock of static electricity. Before touching any electronic components or boards, discharge any static electricity on your

body by using a wrist ground-strap or by touching the bare metal case around the power supply inside your computer. Avoid excessive movement during the installation, such as walking across carpets, as this can generate static. If you must leave the installation area before the installation is complete, be sure to ground yourself again before continuing the installation.

Step 2: Insert Metro-Switch fabric card into the first CPCI 2.16 fabric slot firmly seating it into the Compact PCI chassis. Use ejector handles to secure card into slot so that it is fully inserted and connected to backplane.

Step 3: If using serial port console attach supplied RS232 cable with RJ-11 jack into J1 on Metro-Switch board. Connect other end to serial port DB9 mail connector on PC (COM1, COM2, etc.) and use HyperTerm or another RS232 serial port terminal emulator to connect. Serial port parameters are as follows:

- **8 Data Bits**
- **1 Stop Bit**
- **No Parity**
- **No hardware or software flow control**
- **Speed: 115200 baud**

Step 4: If using external cabling options and/or rear I/O, connect fiber optic cables and CAT5 cables to RJ-45 connectors via Rear I/O.

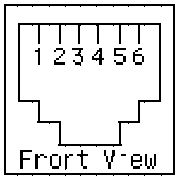
Step 5: After fully seating card and connecting all cables, apply power to system. Green “RUN” LED should begin flashing at the rate of once per every one or two seconds to indicate healthy run status and the system is ready for use.

7.1 SERIAL PORT CABLE DIAGRAM

Included in each Metro-Switch kit shipped from the factory to the customer is one RJ-11 male to DB9 female RS232 serial port cable for connecting to the PC's serial COM port for use with a terminal emulator in a PC. The following table shows the cabling diagram for this serial port cable:

RJ11 Pin	RS232 Signal	COM PORT Signal	DB9 Pin
1	**Not used**	**Not used**	1
2	Transmit Data (TD)	Receive Data (RD)	2
3	Ground	Ground	5
4	**Not used**	**Not used**	4
5	Receive Data (RD)	Transmit Data (TD)	3
6	**Not used**	Ground	6
N/A	**Not used**	**Not used**	7
N/A	**Not used**	**Not used**	8
N/A	**Not used**	**Not used**	9

RS-232 Signals on an RJ11 Jack



RJ11 Wire	Signal	DTE	DCE
1	hardware flow control transmit (optional)	RTS or DTR	CTS and/or DSR
2	transmit		
3	transmit ground		
4	receive ground		
5	receive		
6	hardware flow control receive (optional)	CTS or DSR	RTS and/or DTR

8. COPPER AND FIBER CABLING

8.1. FIBER CABLE SPECIFICATIONS

Distance

(1000-base-SX 850nm multimode)
(1000-base-LX 1310nm singlemode)

1000BASE-SX/LX (850 nm Laser for multimode-SX, 1310nm laser for single-mode-LX)			
Fiber Core Diameter	Type	Fiber Bandwidth Mhz* km	Distance
62.5/125 um	multi-mode	160 Mhz * km	2 to 220 m
62.5/125 um	multi-mode	200 Mhz * km	2 to 275 m
50.0/125 um	multi-mode	400 Mhz * km	2 to 500 m
50.0/125 um	multi-mode	500 Mhz * km	2 to 550 m
9.0/125 um	single-mode	500 Mhz * km	5 km

Connecting fiber optic cable to J2 or J3 ports on Metro-Switch

This section explains how to connect external Ethernet fiber ports to the Metro-Switch when using standard fiber optic cables. Typically 50 or 62.5 micron multimode fiber optic cables with LC type connectors are used for 1000 Base SX operation depending on the connector option. For extended distance, single-mode fiber can be used in models equipped with extended range single-mode connectors (1000 Base LX).

Insert the fiber optic cable into the LC type connector until the self-locking tab clicks into position. Connect the opposite end in to a 1000 Base SX switch. Two types of cables are used when connecting external ports to the Metro-Switch. A workstation or "straight through" cable is typically used to connect Ethernet fiber ports to switches. A fiber “crossover” cable may also be used to connect ports back-to-back. This configuration is useful for loopback and/or diagnostic purposes or when a switch is not available.

8.2 COPPER RJ-45 CONNECTOR AND CAT5 CABLE

Connecting Copper (CAT5/RJ-45) to Rear I/O Transition Module

This section explains how to connect external 1000 Base T copper ports to the Metro-Switch using the standard Category 5, 5e or 6 cables. The maximum cable length is typically 100 meters or 328 feet.

Insert the Category 5 or 5e cable into the RJ-45 connector on the Rear I/O module until the self-locking tab clicks into position. Connect the opposite end in to a device containing a 10/100 or 10/100/1000 Base T port. Two types of cables are used when connecting the Metro-Switch controller to the network. A workstation or "straight through" cable is typically used to connect Ethernet adapters to switches. A “crossover” cable may also be used to connect controllers back-to-back. This configuration is useful for diagnostic purposes or when a hub or switch is not available.

Note(1): The Metro-Switch supports “auto-MDIX” mode where a crossover cable is not required when directly attaching two ports back-to-back.

Note(2)**:** Cables used for Gigabit networks must use all 8 wires. In 10 and 100 modes, wires are dedicated for transmit or receive while in Gigabit mode, data is transmitted and received over all 4-pairs (see pinout diagram below).

RJ-45 pinouts for CAT5 connectors and cables are shown in the following table:

Pin	10/100 Signal	Gigabit Signal
1	Transmit+	Channel A+
2	Transmit-	Channel A-
3	Receive+	Channel B+
4	Unused	Channel C+
5	Unused	Channel C-
6	Receive-	Channel B-
7	Unused	Channel D+
8	Unused	Channel D+

9. UNIT OPERATION

The switch is factory pre-programmed with firmware and default switching configuration to operate as a standalone “layer 2” switch. After power on, the Metro-Switch will initialize and begin to operate a layer 2 switch. This requires no operator intervention and all ports are enabled and will begin to switch traffic based on layer 2 MAC addresses.

9.1 SERIAL PORT CLI COMMAND LINE INTERFACE

The serial port console Command Line Interface (CLI) provides a User Interface for the purpose of local management and is used for Configuration, Status, Statistics and Diagnostic functions. The CLI command line interface is available via the serial port (typically connected to PC and used with terminal emulator window). Configuration functions are provided for general switch parameters, layer 2 switch configuration, layer 3 IP routing table configuration, trunking, mirroring and filtering.

When power is applied to the Metro-Switch and the serial port console is connected and configured, the following message will be displayed:

```
Metro-Switch firmware, ver: 1.12, 12-01-2004
  Copyright (c) 2004 DSS Networks, Inc. All rights reserved.

metro-sw>
```

Configuration ‘set’ commands are stored in eeprom and are loaded and applied to the switch upon power up or internal reset.

NOTE: The CLI commands supported in this firmware release are described in the following sections and include commands to invoke the following capabilities:

- Switch status
- Switch statistics
- Link status
- Link (port) statistics
- Port-based VLANS
- Tagged and untagged VLANS
- Spanning tree groups (MSTP)
- Simple and rapid spanning tree (STP, RSTP)
- Switch reset
- Port statistics
- Configuration commands
- Eeprom access commands

Note: Additional features including trunking, port aggregation, protected ports, IPv4 IP static routing and QoS are planned and will be available in upcoming releases of the switch firmware.

9.1.1 HELP COMMAND

```
metro-sw> h

 CLI cmds:
  link        [show | stat | regs] [int | ext | <arg>] ...
  stats       [show | clr]
  eeprom      [show | init | test] [read | <arg>] <arg>
  eereg       [show | set | clr] <arg> ...
  mmu         regs
  arl         regs
  vlan        [show | set | regs] <arg> ...
  stg         [show] <arg>
  gimask      [show | set | regs] <arg> <arg> ...
  girule      [show | set | regs] <arg> <arg> ...
  gffp-ctr    [show | clr] <arg>
  gffp-pkt    [show | clr] <arg>
  cmic        show
  cfg         [save | clr]
  switch      reset
  gmac        [stat | clr | regs] <arg>
  gpic        regs <arg>
  sgmii       [show | regs] ext <arg>
  portctl     [show | set | regs] <arg> ...
  mirctl      [show | set | regs | clr] <arg> ...
  egrmask     [show | set | regs | clr] <arg> ...
  help
  h
metro-sw>
```

Note: Many commands use a “link index” (linkIdx) for addressing one of 12-ports in the supplied range of 0 to 11.
The **help** command is invoked by either typing ‘**h**’ or ‘**help**’. It lists the available commands with their respective options and arguments. The general command format is ‘**command option <arg> …**’ where ‘**option**’ is one of the options listed above and ‘**<arg> …**’ denotes any number of arguments, including zero. For detailed information on the arguments, please see the corresponding entry below.

9.1.2 VLAN TABLE SET COMMAND

The ***vlan table set*** command can be used to add or delete VLAN table entries specifying member ports. There are two port maps specified. The ‘vlan port map’ bitmask identifies the member ports for this VLAN ID (VID). If the incoming packet is tagged, the VID of the incoming packet is used to index into the VLAN table to get the tagged VLAN port members. If the incoming packet is untagged, then the VID is picked up from the PRTABLE (port default vlan tables) for this port and the VID is used to index into the VLAN table to get the member ports for this VLAN.

All packets inside of the switch will then contain a VID tag. The ‘egr map_untagged’ port bit map specified in the vlan set command simple tells the switch whether to leave the tag on or strip of the tag when transmitting the packet out the egress port.

Usage example:

```
metro-sw> vlan set 3 0x04 0x1ff 0x3ff
```

The arguments to the Vlan set command are as follows:

```
vlan set <id> <stg> <vlan_port_map> <egr_map_untagged>
```

Where:
Id = vlan id
stg = Spanning tree group
vlan_port_map = port bitmap of vlan member ports
egr_map_untagged = port bitmap of egress ports (sent w/o tags)

Note: The port bit map is a 12-bit mask shown as hexadecimal value. For example to enable ports 0, 3 and 11 the mask would be set to: **`0x809 (bits 0, 3 and 11 set).`**

To display VLAN id 3 as set in the example above:

```
metro-sw> vlan show 3

vlan id = 3

 vlan[0x000000]:               9  # value is (stg << 1) | valid_bit
 vlan[0x000001]:             1ff  # egr_mapped_tagged port bitmap
 vlan[0x000002]:             3ff  # egr_mapped_untagged port bitmap

metro-sw>
```

Note(1): To remove an existing VLAN, use the 'vlan set' command specifying both a 'egr_mapped_tagged' and egr_map_untagged with a value of zero as in the following example:

```
vlan set 1 1 0 0
```

Note(2): Once the vlan configurations have been made, you must use the 'cfg save' command to save the configuration to flash eeprom. Changes will take effect upon reset of the switch.

9.1.3 VLAN TABLE SHOW COMMAND

The ***vlan table show*** command can be used to display the VLAN table entries by ***vlan_id*** as shown in the following example:

```
metro-sw> vlan show 3

vlan id = 3

 stg     :               9  # spanning tree group
 port map:             1ff  # egress_mapped_tagged port bitmap
 egr  map:             3ff  # egress_mapped_untagged port bitmap
```

9.1.4 VLAN TABLE REGISTER COMMAND

The vlan table show register command is used to display the value of the corresponding vlan register of the switch e.g., the register value which is used for the current mode of operation.

```
metro-sw> vlan reg 3

vlan id = 03

 vlan[0x3e00000]:                9  # value is (stg << 1) | valid_bit
 vlan[0x3e0000c]:              1ff  # egress_mapped_tagged port bitmap
 vlan[0x3e00018]:              3ff  # egress_mapped_untagged port bitmap
```

9.1.5 CFG SAVE COMMAND

```
metro-sw> cfg save
```

This command is used to save a previously defined configuration to eeprom so that it may be loaded upon issuing a ‘switch reset’ command or upon power up.

Note: The ‘cfg save’ command affect only the switch configuration structure and does not affect the individual ‘eereg set’ commands stored in eeprom. These are initialized and saved independently via the ‘eereg set’ command.

9.1.6 CFG CLEAR COMMAND

```
metro-sw> cfg clr
```

This command is used to clear the configuration records prior to saving to eeprom thereby erasing existing configuration.

Note: This command will clear the configuration and save the values to eeprom immediately. No additional ‘cfg save’ command is necessary.

9.1.7 SWITCH RESET COMMAND

```
metro-sw> switch reset
```

This command is used to initiate a reset (reboot) of the switch.

Note: Any CLI configuration commands entered will be not saved unless a ‘cfg save’ command is first issued prior to reset.

9.1.8 SETTING PACKET AGING

The ‘eereg set’ command is used to set packet aging as in the following example:

```
metro-sw> eereg set 1 0x24 1 1 0x00 0x10 0x00 0x40

In this command, the arguments are interpreted as follows:

metro-sw> eereg set <class=1> <reg-base=0x24> <increment=1> <count=1>
                    <data bytes = 0x00 0x10 0x00 0x40>

metro-sw> eereg show

 reg idx: 01, rclass: 01 (reg set cmd)
      base: 24,  strt/incr: 01, end/cnt: 01
      data: 00 10 00 40
```

In the example above, the last two data bytes (0x00 0x40) specify the layer 2 ARL age timeout in seconds (i.e. 0x00 0x40 hexidecimal equals 64 seconds).

9.1.9 SETTING DEFAULT VLAN

The ‘eereg set’ command is used to set the default VLAN id for a port or range of ports as in the following example:

```
metro-sw> eereg set 3 0x21 0 1 0x00 0x00 0x00 0x01

In this command, the arguments are interpreted as follows:

metro-sw> eereg set <class=3> <reg-base=0x21> <strt-port=0> <end-port=1>
                    <data bytes = 0x00 0x00 0x00 0x01>

metro-sw> eereg show

 reg idx: 02, rclass: 03 (reg set cmd)
      base: 21,  strt/incr: 00, end/cnt: 01
      data: 00 00 00 01
```

In the example above, the last two data bytes (0x00 0x01) specify the default VLAN id for the port range starting at zero and ending at 1 (port 0 and port1). Any port range (i.e. 0 – 11) may be specified with this command.

9.1.10 SETTING UNTAGGED VLAN ID’s

The ‘eereg set’ command is used to set up to seven (7) default VLAN ID’s per port. This command can operate on a single port or range of ports as in the following example:

```
metro-sw> metro-sw> eereg set 3 0x22 0 0 0x01 0x08 0x00 0x01

In this command, the arguments are interpreted as follows:

metro-sw> eereg set <class=3> <reg-base=0x22> <strt-port=0> <end-port=0>
                    <data bytes = 0x01 0x08 0x00 0x01>
```

Note: ‘reg-base’ can be set from 0x22 – 0x28 for up to seven entries per port.

```
metro-sw> eereg show

 reg idx: 01, rclass: 03 (reg set cmd)
      base: 22,  strt/incr: 00, end/cnt: 00
      data: 01 08 00 01
```

In the example above, the data bytes comprising a 32-bit long word (0x01, 0x08, 0x00 0x01) specify the VLAN id, Ethertype and Frame-Type for the port range starting at zero and ending at 0 (port 0 only).

The following table shows the encoding of the data bytes comprising the 32-bit longword argument:

Bits	**Name**	Description
30:28	FrameType 001 = Ethernet II 010 = 802.3 LLC/SNAP 100 = LLC (Only one bit set per entry)	Frame type to match
Bits	**Name**	Description
27:12	EtherType (i.e. 0x0800 for IPv4)	Ether type field which determines the protocol
Bits	**Name**	Description
11:00	VLAN ID	Vlan ID for this entry

9.1.11 'EEREG SET' COMMAND

The 'eereg set' command is used to set and enable certain features of the switch functionality. For example:

```
metro-sw> eereg set 1 0x24 1 1 00 10 00 40
```

The parameters for the eereg set command are interpreted as follows:

```
metro-sw> eereg set <reg-class> <reg-offset> <strt/incr> <end/cnt> <quad-bytes>
```

To show (display) all existing 'eereg set' entries:

```
metro-sw> eereg show

 reg idx: 01, rclass: 01 (reg set cmd)
      base: 24,  strt/incr: 01, end/cnt: 01
      data: 00 10 00 40
```

To clear out (erase) all existing 'eereg set' entries:

```
metro-sw> eereg set 0
```

Note: 'eereg set' commands are provided as a basic capability to set and enable certain features of the switch functionality. Please contact our technical support department regarding the setting of any additional features not shown in this document. The switch must be reset after all reg-set commands have been entered in order to take effect.

9.1.12 ‘EEREG CLR’ COMMAND

```
metro-sw> eereg clr <idx>
```

Clears eeprom entries specified by ‘reg-idx’.

9.1.13 ‘EEREG SHOW’ COMMAND

```
metro-sw> eereg show
```

The ‘eereg show’ command displays all eeprom reg-set entries currently stored as in the following example:

```
metro-sw> eereg show

 reg idx: 01, rclass: 03 (reg set cmd)
      base: 22,  strt/incr: 00, end/cnt: 00
      data: 01 08 00 01


 reg idx: 02, rclass: 01 (reg set cmd)
      base: 24,  strt/incr: 01, end/cnt: 01
      data: 00 10 00 40


 reg idx: 03, rclass: 03 (reg set cmd)
      base: 21,  strt/incr: 00, end/cnt: 01
      data: 00 00 00 01
```

9.1.14 LINK COMMAND

```
metro-sw> link show int 0xb

 phy link = 11, link up = 1
 reg(0x0):      1140
 reg(0x1):      796d
 reg(0x1c-68):  69bf
 reg(0x1c-7c):  7cec

metro-sw> link show ext 4

 iphy link =  4, link up = 1
 reg(0x01): 0004         reg(0x00): 0140
 reg(0x05): 0000         reg(0x06): 0000
 reg(0x0b): 0004         reg(0x0c): 180c
 reg(0x0e): 0000         reg(0x0f): 0000
 reg(0x10): 0400         reg(0x14): 001d
```

The ‘link stat’ command shows a summary table of the link state (down/up) and link speed.

An argument ‘all’ or ‘-1’ shows a brief summary of all links. A numeric argument between 0-11 shows a summary of the specified link.

Usage example:

```
metro-sw> linkst all

  Link   Up   Spd
  ----   --   ----
    0     1   1000
    1     1   1000
    2     0      0
    3     0      0
    4     1   1000
    5     0      0
    6     0      0
    7     0      0
    8     0      0
    9     0      0
   10     1   1000
   11     1   1000

metro-sw> link stat 4

 phy link =  4, link up = 1, speed =  1000
  link  state:     1
  ed    state:     1
  sd    state:     0
  sgmii state:     1
  mac   state:     1
  lk down cnt:     5
  lk up   cnt:     1
  lk cfg  cnt:     2
  ed down cnt:     0
  ed up   cnt:     1
  sd down cnt:     0
```

```
  sd up   cnt:     0
  an strt cnt:     0
```

9.1.15 LINK REGS COMMAND

```
metro-sw> link regs int 0

 iphy[0x00]:     140
 iphy[0x01]:       4
 iphy[0x02]:       0
 iphy[0x03]:       0
 iphy[0x04]:      40
 iphy[0x05]:       0
 iphy[0x06]:       0
 iphy[0x07]:       0
 iphy[0x08]:       0
 iphy[0x09]:       0
 iphy[0x0a]:       0
 iphy[0x0b]:       4
 iphy[0x0c]:    180c
 iphy[0x0d]:       0
 iphy[0x0e]:       0
 iphy[0x0f]:       0
 iphy[0x10]:     400
 iphy[0x11]:     330
 iphy[0x12]:    3fff
 iphy[0x13]:       0
 iphy[0x14]:      1d
 iphy[0x15]:       2
 iphy[0x16]:     100
 iphy[0x17]:       0
 iphy[0x18]:       0
 iphy[0x19]:       0
 iphy[0x1a]:       0
 iphy[0x1b]:       0
 iphy[0x1c]:       0
 iphy[0x1d]:       0
 iphy[0x1e]:       0
 iphy[0x1f]:       0

metro-sw> link regs ext 2

 phy[0x00]:     1140
 phy[0x01]:     796d
 phy[0x02]:       20
 phy[0x03]:     60b1
 phy[0x04]:      1e1
 phy[0x05]:     cde1
 phy[0x06]:        f
 phy[0x07]:     2001
 phy[0x08]:     402d
 phy[0x09]:      700
 phy[0x0a]:     7c00
 phy[0x0b]:        0
 phy[0x0c]:        0
 phy[0x0d]:        0
 phy[0x0e]:        0
 phy[0x0f]:     3000
 phy[0x10]:        1
 phy[0x11]:     2303
```

```
phy[0x12]:        0
phy[0x13]:        0
phy[0x14]:      101
phy[0x15]:    60e6
phy[0x16]:        0
phy[0x17]:     f03
phy[0x18]:    7267
phy[0x19]:    ff1c
phy[0x1a]:    247e
phy[0x1c]:    7cec
phy[0x1d]:     57b
phy[0x1e]:        0
phy[0x1f]:        0
s-1c[0x34]:   3421
s-1c[0x38]:   3868
s-1c[0x68]:   68c0
s-1c[0x70]:   721c
s-1c[0x78]:   7927
s-1c[0x7c]:   7cec
xphy[0x00]:   ffff
xphy[0x01]:     60
xphy[0x02]:     ff
xphy[0x03]:   60e6
xphy[0x05]:    1c1
xphy[0x07]:   6de7
xphy[0x08]:    bdf
xphy[0x09]:   3200
xphy[0x0a]:   13ff
xphy[0x0b]:   69e6
xphy[0x0c]:    190
xphy[0x0d]:    bd5
xphy[0x0e]:    320
xphy[0x0f]:   ffef
```

9.1.16 GMAC STATS SHOW COMMAND

The switch provides a large number of statistical counters to support system management capabilities and include information to support RMON, SNMP and Ethernet MIBs as shown below:

```
metro-sw> gmac stat 0

link:  2 mib stats

 offset  name        count
 ------  --------- -------
  (0x20)  GTR64           8  (1757/EtherStatsPkts64Octets)
  (0x21)  GTR127      19078  (1757/EtherStatsPkts65to127Octets)
  (0x22)  GTR255          0  (1757/EtherStatsPkts128to255Octets)
  (0x23)  GTR511          0  (1757/EtherStatsPkts256to511Octets)
  (0x24)  GTR1023         0  (1757/EtherStatsPkts512to1023Octets)
  (0x25)  GTR1518     19080  (1757/EtherStatsPkts1024to1518Octets)
  (0x26)  GTRMGV          0
  (0x27)  GTR2047         0
  (0x28)  GTR4095         0
  (0x29)  GTR9216         0
  (0x2c)  GRPKTs      19090  (1757/EtherStatsPkts;1493/dot1dTpPortInFrames)
  (0x2d)  GRUND*          0  (1757/EtherStatsUndersizePkts)
  (0x2e)  GRFRG           0  (1757/EtherStatsFragments)
  (0x2f)  GRBYTes  15116368  (1757/EtherStatsOctets)
  (0x30)  GRMCast         0  (1757/EtherStatsMulticastPkts)
  (0x31)  GRBCast         0  (1757/EtherStatsBroadcastPkts)
  (0x32)  GRFCS*          0  (1757/EtherStatsCRCAlignErrs;1643/dot3StatsFCSErrs)
  (0x33)  GRXCF           0  (802.2-30.3.3.4/aMACCtrlFrmsRcvd)
  (0x34)  GRXPF           0  (802.2-30.3.4.3/aMACPauseCtrlFrmRcvd;2665/dot3InPauseFrames)
  (0x35)  GRXUO*          0  (802.2-30.3.3.5/aUnsuppOpcRcvd;2665/dot3CtrlUnkOpcodes)
  (0x36)  GRALN*          0
  (0x37)  GRFLR*          0
  (0x38)  GRCDE*          0
  (0x39)  GRFCR*          0
  (0x3a)  GROVR*          0
  (0x3b)  GRJBR*          0
  (0x3c)  GRIPC           0
  (0x3d)  GIMBP           0
  (0x3e)  GIMRP           0
  (0x3f)  GRIMDR          0
  (0x40)  GRIPD           0
  (0x41)  GRIPHE*         0
  (0x42)  GRDISC          0
  (0x43)  GRUC        19104
  (0x44)  GPDISC          0
  (0x45)  GRFILDR         0
  (0x46)  GRPORTD         0
  (0x48)  GTPKTs      19108  (1493/dot1dTpPortOutFrames)
  (0x49)  GTMCast         0  (802.2-30.3.1.18/aMulticastXmittedOk)
  (0x4a)  GTBCast         0  (802.2-30.3.1.19/aBroadcastXmittedOk)
  (0x4b)  GTXPF           0
  (0x4c)  GTJBR           0
  (0x4d)  GTFCS*          0
  (0x4e)  GTXCF           0
  (0x4f)  GTOVR*          0
  (0x50)  GTDFR           0
  (0x51)  GTEDF*          0
  (0x52)  GTSCL*          0
```

```
  (0x53)  GTMCL*         0
  (0x54)  GTLCL*         0
  (0x55)  GTXCL*         0
  (0x56)  GTFrags        0
  (0x57)  GTNCL*         0
  (0x5b)  GTBYTes  5138544  (802.2-30.3.1.1.8/aOctetsTransmittedOk;2665/ifOutOctets)
  (0x5c)  GTIP           0
  (0x5d)  GTVLAN         0
  (0x5e)  GTAGE          0
  (0x5f)  GTIPD*         0
  (0x60)  GTIMTLD        0
  (0x61)  GTABRT*        0
  (0x62)  GTIMDR         0
  (0x63)  GTFIDR         0
  (0x74)  GTCE*          0
  (0x75)  GTIPAGE        0
```

9.1.17 GMAC STATS CLEAR COMMAND

The gmac stats clear command executes an implicit ‘gmac stat <port>’ command before clearing the stats values for the selected port.

```
metro-sw> gmac clr 3
```

9.1.18 GMAC REGS SHOW COMMAND

```
metro-sw> gmac regs 11

 gmac[0x0000]:         1
 gmac[0x0001]:  550a1681
 gmac[0x0002]:         c
 gmac[0x0003]:         6
 gmac[0x0004]:   180c200
 gmac[0x0005]:     10000
 gmac[0x0008]:      2328
 gmac[0x0100]:         1
 gmac[0x0101]:      4101
 gmac[0x0102]:        15
 gmac[0x0103]:       c12
 gmac[0x0104]:      370f
 gmac[0x0105]:      23ff
 gmac[0x0110]:         0
 gmac[0x0111]:         0
 gmac[0x0112]:         0
```

9.1.19 GENERAL STATS COMMAND

```
metro-sw> stats show
  ms: unit:     0 statistics
  main loop cnt                 7888
  sw i2c read cnt              63949
  sw i2c read fail                 0
  sw i2c write cnt            127405
  sw i2c write fail                0
  sw ack cnt                 1515882
  sda val                          0
  wait clk high                    0
  wait clk low                     0
```

```
  wait data high                  0
  last read val                   0
  last progress code              4
  mii timeouts                    0
  schan timeouts                  0
  schan err cnt                   0
  last schan rsp                  0
  l2 purge cnt                    0
```

The clear stats command executes an implicit ‘stats show’ command before clearing the stats values.

```
metro-sw> stats clr
  ms: unit:     0 statistics
  main loop cnt                8976
  sw i2c read cnt             72765
  sw i2c read fail                0
  sw i2c write cnt           144969
  sw i2c write fail               0
  sw ack cnt                1724862
  sda val                         0
  wait clk high                   0
  wait clk low                    0
  wait data high                  0
  last read val                   0
  last progress code              4
  mii timeouts                    0
  schan timeouts                  0
  schan err cnt                   0
  last schan rsp                  0
  l2 purge cnt                    0
```

9.1.20 GPIC REGS SHOW COMMAND

```
metro-sw> gpic regs 0

 gpic[0x80000]:  1001800
 gpic[0x80001]:        1
 gpic[0x80002]:     1001
 gpic[0x80003]:      fff
 gpic[0x80004]:        0
 gpic[0x80005]:        0
 gpic[0x80006]:        0
 gpic[0x80007]:        0
 gpic[0x80008]:        0
 gpic[0x80009]: c2000020
 gpic[0x8000a]:      180
 gpic[0x8000b]: c2000021
 gpic[0x8000c]:      180
 gpic[0x8000d]:        0
 gpic[0x8000e]:        0
 gpic[0x8000f]:        0
 gpic[0x80010]:        0
 gpic[0x80011]:        0
 gpic[0x80012]:        0
 gpic[0x80013]:        0
 gpic[0x80014]:        0
 gpic[0x80015]:        0
 gpic[0x80016]:        0
 gpic[0x80017]:        0
```

```
 gpic[0x80018]:        0
 gpic[0x80019]:        0
 gpic[0x8001a]:        0
 gpic[0x8001b]:        0
 gpic[0x8001c]:        0
 gpic[0x8001d]:        0
 gpic[0x8001e]:        0
 gpic[0x8001f]:        0
 gpic[0x80020]:        0
 gpic[0x80021]:        1
 gpic[0x80022]:        0
 gpic[0x80023]:        0
 gpic[0x80024]:        0
 gpic[0x80025]:        0
 gpic[0x80026]:        0
 gpic[0x80027]:        0
 gpic[0x80028]:        0
```

9.1.21 SGMII REGS SHOW COMMAND

```
metro-sw> sgmii regs ext 11

 sgmii[0x   0]:  140
 sgmii[0x   1]:  149
 sgmii[0x   2]:   20
 sgmii[0x   3]: 60b1
 sgmii[0x   4]:    1
 sgmii[0x   5]:    0
 sgmii[0x   6]:    2
 sgmii[0x   7]:    0
 sgmii[0x   8]:    0
 sgmii[0x   9]:    0
 sgmii[0x   a]:    0
 sgmii[0x   b]:    0
 sgmii[0x   c]:    0
 sgmii[0x   d]:    0
 sgmii[0x   e]:    0
 sgmii[0x   f]: c000
 sgmii[0x  10]:    0
 sgmii[0x  11]: 2000
 sgmii[0x  12]:    0
 sgmii[0x  13]:    0
 sgmii[0x  14]:    0
 sgmii[0x  15]: e0e7
 sgmii[0x  16]:    0
 sgmii[0x  17]:  f03
 sgmii[0x  18]:  400
 sgmii[0x  19]: 1000
 sgmii[0x  1a]:    0
 sgmii[0x  1c]: 7c1d
 sgmii[0x  1d]:    0
 sgmii[0x  1e]:    0
 sgmii[0x  1f]:    0
```

9.1.22 SGMII STATUS SHOW COMMAND

```
metro-sw> sgmii show ext 11

 sgmii phy link = 11, link up = 0
```

```
 reg(0x0):      000b
 reg(0x1):      0149
 reg(0x1c-68):  69bf
 reg(0x1c-7c):  7c1d

metro-sw> sgmii show ext 0

 sgmii phy link =  0, link up = 0
 reg(0x0):      0000
 reg(0x1):      0149
 reg(0x1c-68):  68c0
 reg(0x1c-7c):  7ced
```

9.1.23 EEPROM CONTENTS SHOW COMMAND

```
metro-sw> eeprom show 0 20

 offset: 0000, cnt:   20
0000: fe fe fe fe fe fe fe fe - fe fe fe fe fe fe fe fe
0016: fe fe fe fe

metro-sw> eeprom init -1 253
metro-sw> eeprom init 0 256
metro-sw> eeprom show 0 64

 offset: 0000, cnt:   64
0000: fd fd fd fd fd fd fd fd - fd fd fd fd fd fd fd fd
0016: fd fd fd fd fd fd fd fd - fd fd fd fd fd fd fd fd
0032: fd fd fd fd fd fd fd fd - fd fd fd fd fd fd fd fd
0048: fd fd fd fd fd fd fd fd - fd fd fd fd fd fd fd fd
```

9.1.24 EPROM INIT COMMAND

```
metro-sw> eeprom init -1 250 # sets pattern for initialization to value=250

metro-sw> eeprom init 0 256  # offset=0, count=256

 eeprom init done.
```

9.1.25 EEPROM TEST READ COMMAND

```
metro-sw> eeprom test read 100

 eeprom test read begin.
 eeprom test read done.
```

9.1.26 MMU REGS SHOW COMMAND

```
metro-sw> mmu regs 0

 mmu[0xd80000]:  5555555
 mmu[0xd80001]:  5555555
 mmu[0xd80002]:        0
 mmu[0xd80003]:        0
 mmu[0xd80004]:  5555555
```

```
 mmu[0xd80005]:  5555555
 mmu[0xd80006]:        0
 mmu[0xd80007]:        0
 mmu[0xd80008]:        0
 mmu[0xd80009]:        0
 mmu[0xd8000a]:        0
 mmu[0xd8000b]:        0
 mmu[0xd8000c]:        0
 mmu[0xd8000d]:        0
 mmu[0xd8000e]:      280
 mmu[0xd8000f]:      280
 mmu[0xd80010]:      280
 mmu[0xd80011]:      280
 mmu[0xd80012]:      280
 mmu[0xd80013]:      280
 mmu[0xd80014]:      280
 mmu[0xd80015]:      280
 mmu[0xd80016]:    84210
 mmu[0xd80017]:    84210
 mmu[0xd80018]:     1fff
 mmu[0xd80019]:       30
 mmu[0xd8001a]:     1f00
 mmu[0xd8001b]:     1e00
 mmu[0xd8001c]:     3518
 mmu[0xd8001d]:     a237
 mmu[0xd8001e]:     3fff
 mmu[0xd8001f]:     3fff
 mmu[0xd80020]:        0
 mmu[0xd80021]:     1fff
 mmu[0xd80022]:        0
```

9.1.27 CMIC REGS SHOW COMMAND

```
metro-sw> cmic show

 reg(0x0178):    35690
 reg(0x0050):    48002
 reg(0x010c):  4a00010
 reg(0x0140):     3fff
 reg(0x0160):     3fff
 reg(0x0158): 1c0bb4df
 reg(0x015c):     34df
 reg(0x0164): ffefffff
 reg(0x0168):        0
 reg(0x016c):     3fff
 reg(0x0120):       4c
 reg(0x0128):       4c
 reg(0x012c):       4c
```

9.1.28 ARL REGS SHOW COMMAND

```
metro-sw> metro-sw> arl regs

 arl[0xe80000]:        0
 arl[0xe80001]:        0
 arl[0xe80002]:        0
 arl[0xe80003]:        0
 arl[0xe80004]:        0
 arl[0xe80005]:        0
 arl[0xe80006]:        0
 arl[0xe80007]:        0
 arl[0xe80008]:        0
 arl[0xe80009]:        0
 arl[0xe8000a]:        0
 arl[0xe8000b]:        0
 arl[0xe8000c]:        0
 arl[0xe8000d]:        0
 arl[0xe8000e]:        0
 arl[0xe8000f]:        0
 arl[0xe80010]:        0
 arl[0xe80011]:        0
 arl[0xe80012]:        0
 arl[0xe80013]:        0
 arl[0xe80014]:        0
 arl[0xe80015]:        0
 arl[0xe80016]:        0
 arl[0xe80017]:        0
 arl[0xe80018]:        0
 arl[0xe80019]:        0
 arl[0xe8001a]:        0
 arl[0xe8001b]:        0
 arl[0xe8001c]:        0
 arl[0xe8001d]:        0
 arl[0xe8001e]:        0
 arl[0xe8001f]:        0
 arl[0xe80020]:        0
 arl[0xe80021]:        0
 arl[0xe80022]:        0
 arl[0xe80023]:        0
 arl[0xe80024]:   100040
 arl[0xe80025]:        0
 arl[0xe80026]:        0
 arl[0xe80027]:        0
 arl[0xe80028]:        e
```

9.1.29 STG SHOW COMMAND

The ‘stg show’ command shows the spanning tree groups and default state specified by group index (default state = **`0x3ffffff`** if initialized):

```
metro-sw> stg show 1

stg id = 01

 stg[0x3e10000]:        3ffffff
```

9.1.30 'GIMASK SET' COMMAND

The 'gimask set' command is used to setup the GIMASK tables for the FFP processor. The command takes seven numeric arguments. The first argument is the 'link' index (0-11) and the second argument is the 'mask' index (0-15). The third argument is the table entry index (0 – 11). The last 4 arguments are the data bytes comprising the 32-bit dword. The rightmost argument 'd1' is the least significant byte (bits 0 – 7) while the leftmost 'd3' is the most significant byte (bits 24-31) of the 32-bit dword.

```
metro-sw> gimask set <linkIdx> <maskIdx> <entryIdx> <0xd4 0xd3 0xd2 0xd1>
```

Usage examples:

The following example shows setting up the FFP IMASK and IRULE tables for a simple method of filtering, re-routing and mirroring. Ports 0 and 4 are connected to external PC's and there is a ping traffic running into port 4 from one PC going to the second PC connected to port 0. The ping reply goes in the opposite direction.

The example filters the packets arriving at port 0 and if a match is found sends the packets out port 1. In addition, the egress of port 0 (going out port 1) is mirrored to port 4 resulting in the original ping traffic being returned to the source port 4 as well as being sent out port 1.

The FFP IMASK and IRULE configuration below sets up the FFP to filter the Ethernet II IP frame type of '0x0800' with the first word of the IP header of '0x4500' which is found at offset 12 in the packet header. In the FFP matching, it occurs at offset 16 which is probably due to packet type normalization.

The example shown sets up one mask and one rule entry to filter the four bytes of 08 00 45 00. The action in the IRULE instructs the FFP to send the packet out and alternate port as specified in the rule table. A 'GFFPCOUNTER' is also instructed to be incremented for each action taken (packet send to port 1). The comments next to the following commands indicate which settings are taking place.

```
# GIMASK settings

metro-sw> gimask set 0 0 0 0 0x80 0 0          # set rules-size=1 and start-index=0
  midx: 00, eidx: 00

metro-sw> gimask set 0 0 1 0 0 0 0x18          # set DATAOFF-2 = 3 (bytes 16-19)
  midx: 00, eidx: 01

metro-sw> gimask set 0 0 2 0 0x2 0 0           # set the 'untag' bit
  midx: 00, eidx: 02

metro-sw> gimask set 0 0 5 0xff 0 0xff 0       # set FMASK-2 (for dataoff-2 mask)
  midx: 00, eidx: 05

# GIRULE settings
```

```
metro-sw> girule set 0 0 0 0 0 0 0xa0          # set action bits 5 and 7 (send to
port)
  midx: 00, eidx: 00                           # (and increment counter)

metro-sw> girule set 0 0 2 0 0 0 0x21          # dest-port=1, untag-bit, ctr=0
  midx: 00, eidx: 02

metro-sw> girule set 0 0 5 0x08 0 0x45 0       # set FILTER-2 (for dataoff-2 filter)
  midx: 00, eidx: 05

metro-sw> eereg set 3 5 0 0 0 0x5 0 0xff       # enable mirroring on port 0 egress
port 1

metro-sw> cfg save                             # save configuration to eeprom
```

FFP “GIMASK” table bit map description:

ADDR-LIMIT 0x07g0.000B:table bit 383

FMASK(8)																															
FMASK(7)																															
FMASK(6)																															
FMASK(5)																															
FMASK(4)																															
FMASK(3)																															
FMASK(2)																															
FMASK(1)																															
RSVD									OUTPUTMODMASK(5)					RSVD(7)							EGRMASK(5)					PKTFORMAT MASK(4)			
RSVD									OUTPUTMOD(5)					U	OUTPUTPORT(5)						TOS_P(3)			DIFFSERV(6)					IEEE802-DOT1PRI		
NOTDEFINED									DATAOFF-8					DATASOFF=7						DATAOFF-6			DATAOFF-5			DATAOFF-4			DATAOFF-3			DATAOFF-2			DATAOFF-1		
R		RULES-SIZE(8)							RULES-START(7)									NOMATCHACTION(16)															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

ADDR-BASE 0x07g0.0000:table bit 0

Additional notes: See the following Broadcom documents for further details regarding the programming of the FFP:

Programmer's Reference Guide	569x-PG101-R	StrataXGS(tm) BCM5690/BCM5691/BCM5692/BCM5693
Application Note	5690-AN100-R	BCM5690 Fast Filtering Processor Application Note
Application Note	5690-AN900-R	BCM5690 Application Note - StrataXGS - Traffic Mirroring

9.1.31 ‘GIRULE SET’ COMMAND

The ‘girule set’ command is used to setup the “GIRULE” tables for the FFP processor. The command takes seven numeric arguments. The first argument is the ‘link’ index (0-11) and the second argument is the ‘rule’ index (0-63). The third argument is the table entry index (0 – 11). The last 4 arguments are the data bytes comprising the 32-bit

dword. The rightmost argument ‘d1’ is the least significant byte (bits 0 – 7) while the leftmost ‘d3’ is the most significant byte (bits 24-31) of the 32-bit dword.

```
metro-sw> girule set <linkIdx> <ruleIdx> <entryIdx> <0xd4 0xd3 0xd2 0xd1>
```

Usage examples:

```
metro-sw> girule set 0 0 0 0 0 0 0xa0
  midx: 00, eidx: 00

metro-sw> girule set 0 0 2 0 0 0 0x21
  midx: 00, eidx: 02

metro-sw> girule set 0 0 5 0x08 0 0x45 0
  midx: 00, eidx: 05

metro-sw> cfg save
```

FFP “GIRULE” table bit map description:

ADDR-LIMIT 0x07g2.000B:table bit 383

FILTER(8)																															
FILTER(7)																															
FILTER(6)																															
FILTER(5)																															
FILTER(4)																															
FILTER(3)																															
FILTER(2)																															
FILTER(1)																															
RSVD(7)							PRI(3)			TOS_P(3)			FSEL(4)			EGRSMOD(5)															
RSVD(4)				METERID(6)						COUNTER(5)			DIFFSERV(6)						OUTPUT MOD 5			U			DST_PORT(5)					
RSVD(5)								CLASSIFICATION-TAG(16)												OUT_DSCP(6)						OUTACTIONS(5)					
R		VLANID(12)												ACTION(18)																	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

ADDR-BASE 0x07g2.0000:table bit 0

9.1.32 ‘GIMASK SHOW’ COMMAND

The ‘gimask show’ command is used to display the GIMASK tables for the FFP processor. The command takes two numeric arguments. The first argument is the ‘link’ index (0-11) and the second argument is the ‘mask’ index (0-63).

If <linkIdx> is either ‘-1’ or ‘all’, all valid entries in the table are shown.

If <maskIdx> is either ‘-1’ or ‘all’, all valid table entries with the specified link index are shown.

```
metro-sw> gimask show <linkIdx> <maskIdx>
```

Usage examples:

```
metro-sw> gimask show 0 0

ffpm link: 00, midx: 00

 ffpm[0x7000000]:         800000
 ffpm[0x7000001]:             18
 ffpm[0x7000002]:          20000
 ffpm[0x7000003]:              0
 ffpm[0x7000004]:              0
 ffpm[0x7000005]:       ff00ff00
 ffpm[0x7000006]:              0
 ffpm[0x7000007]:              0
 ffpm[0x7000008]:              0
 ffpm[0x7000009]:              0
 ffpm[0x700000a]:              0
 ffpm[0x700000b]:              0
```

9.1.33 ‘GIRULE SHOW’ COMMAND

The ‘girule show’ command is used to show GIMASK tables for the FFP processor. The command takes two numeric arguments. The first argument is the ‘link’ index (0-11) and the second argument is the ‘rule’ index (0-63).

If <linkIdx> is either ‘-1’ or ‘all’, all valid entries in the table are shown.

If <ruleIdx> is either ‘-1’ or ‘all’, all valid table entries with the specified link index are shown.

```
metro-sw> girule show <linkIdx> <ruleIdx>
```

Usage example:

```
metro-sw> girule show 0 0

ffpr link: 00, ridx: 00

 ffpr[0x7020000]:             a0
 ffpr[0x7020001]:              0
 ffpr[0x7020002]:             21
 ffpr[0x7020003]:              0
 ffpr[0x7020004]:              0
 ffpr[0x7020005]:        8004500
 ffpr[0x7020006]:              0
```

```
 ffpr[0x7020007]:              0
 ffpr[0x7020008]:              0
 ffpr[0x7020009]:              0
 ffpr[0x702000a]:              0
 ffpr[0x702000b]:              0
```

9.1.34 ‘GIMASK SHOW REGISTER’ COMMAND

The ‘gimask regs’ command is used to display the current GIMASK tables for the FFP processor currently used by the switch. The command takes two numeric arguments. The first argument is the ‘link’ index (0-11) and the second argument is the ‘mask’ index (0-63).

Note: By default, the tables are not initialized

```
metro-sw> gimask regs <linkIdx> <maskIdx>
```

Usage examples:

```
metro-sw> gimask regs 0 0

ffpm link: 00, midx: 00

 ffpm[0x7000000]:         800000
 ffpm[0x7000001]:             18
 ffpm[0x7000002]:          20000
 ffpm[0x7000003]:              0
 ffpm[0x7000004]:              0
 ffpm[0x7000005]:       ff00ff00
 ffpm[0x7000006]:              0
 ffpm[0x7000007]:              0
 ffpm[0x7000008]:              0
 ffpm[0x7000009]:              0
 ffpm[0x700000a]:              0
 ffpm[0x700000b]:              0
```

9.1.35 ‘GIRULE SHOW REGISTER’ COMMAND

The ‘girule regs’ command is used to display the GIMASK tables for the FFP processor currently used by the switch. The command takes two numeric arguments. The first argument is the ‘link’ index (0-11) and the second argument is the ‘rule’ index (0-63).
Note: By default, the tables are not initialized.

```
metro-sw> girule regs <linkIdx> <ruleIdx>
```

Usage example:

```
metro-sw> girule regs 0 0

ffpr link: 00, ridx: 00

 ffpr[0x7020000]:             a0
```

```
 ffpr[0x7020001]:              0
 ffpr[0x7020002]:             21
 ffpr[0x7020003]:              0
 ffpr[0x7020004]:              0
 ffpr[0x7020005]:        8004500
 ffpr[0x7020006]:              0
 ffpr[0x7020007]:              0
 ffpr[0x7020008]:              0
 ffpr[0x7020009]:              0
 ffpr[0x702000a]:              0
 ffpr[0x702000b]:              0
```

9.1.36 ‘GFFP-CTR’ SHOW COMMAND

The ‘gffp-ctr show’ command is used to display the ffp counters for a particular link. The link index is used as an argument.

```
metro-sw> gffp-ctr show <linkIdx>
```

Usage example:

```
metro-sw> gffp-ctr show 0

gffp ctrs link: 00

 gffp-ctr[0x7050000]:       14df
 gffp-ctr[0x7050004]:          0
 gffp-ctr[0x7050008]:          0
 gffp-ctr[0x705000c]:          0
 gffp-ctr[0x7050010]:          0
 gffp-ctr[0x7050014]:          0
 gffp-ctr[0x7050018]:          0
 gffp-ctr[0x705001c]:          0
 gffp-ctr[0x7050020]:          0
 gffp-ctr[0x7050024]:          0
 gffp-ctr[0x7050028]:          0
 gffp-ctr[0x705002c]:          0
 gffp-ctr[0x7050030]:          0
 gffp-ctr[0x7050034]:          0
 gffp-ctr[0x7050038]:          0
 gffp-ctr[0x705003c]:          0
 gffp-ctr[0x7050040]:          0
 gffp-ctr[0x7050044]:          0
 gffp-ctr[0x7050048]:          0
 gffp-ctr[0x705004c]:          0
 gffp-ctr[0x7050050]:          0
 gffp-ctr[0x7050054]:          0
 gffp-ctr[0x7050058]:          0
 gffp-ctr[0x705005c]:          0
 gffp-ctr[0x7050060]:          0
 gffp-ctr[0x7050064]:          0
 gffp-ctr[0x7050068]:          0
 gffp-ctr[0x705006c]:          0
 gffp-ctr[0x7050070]:          0
 gffp-ctr[0x7050074]:          0
 gffp-ctr[0x7050078]:          0
 gffp-ctr[0x705007c]:          0
```

9.1.37 ‘GFFP-CTR’ CLEAR COMMAND

The ‘gffp-ctr clr’ command is used to clear the ffp counters for the specified link. The link index is used as an argument.
Note: The ‘gffp-ctr clr’ command issues an implicit ‘gffp-ctr show’ command before clearing the counters.

```
metro-sw> gffp-ctr clr <linkIdx>
```

9.1.38 ‘GFFP-PKT’ CLEAR COMMAND

The ‘gffp-pkt clr’ command is used to clear the ffp packet counters for the specified link. The link index is used as an argument.
Note: The ‘gffp-pkt clr’ command issues an implicit ‘gffp-pkt show’ command before clearing the counters.

```
metro-sw> gffp-pkt clr <linkIdx>
```

9.1.39 ‘GFFP-PKT’ SHOW COMMAND

The ‘gffp-pkt show’ command is used to display the ffp packet counters for a particular link. The link index is used an argument.

```
metro-sw> gffp-pkt show <linkIdx>
```

Usage example:

```
metro-sw> gffp-pkt show 0

gffp pkt ctrs link: 00

 gffp-pkt[0x7040000]:          0
 gffp-pkt[0x7040004]:          0
 gffp-pkt[0x7040008]:          0
 gffp-pkt[0x704000c]:          0
 gffp-pkt[0x7040010]:          0
 gffp-pkt[0x7040014]:          0
 gffp-pkt[0x7040018]:          0
 gffp-pkt[0x704001c]:          0
 gffp-pkt[0x7040020]:          0
 gffp-pkt[0x7040024]:          0
 gffp-pkt[0x7040028]:          0
 gffp-pkt[0x704002c]:          0
 gffp-pkt[0x7040030]:          0
 gffp-pkt[0x7040034]:          0
 gffp-pkt[0x7040038]:          0
 gffp-pkt[0x704003c]:          0
 gffp-pkt[0x7040040]:          0
 gffp-pkt[0x7040044]:          0
```

```
gffp-pkt[0x7040048]:          0
gffp-pkt[0x704004c]:          0
gffp-pkt[0x7040050]:          0
gffp-pkt[0x7040054]:          0
gffp-pkt[0x7040058]:          0
gffp-pkt[0x704005c]:          0
gffp-pkt[0x7040060]:          0
gffp-pkt[0x7040064]:          0
gffp-pkt[0x7040068]:          0
gffp-pkt[0x704006c]:          0
gffp-pkt[0x7040070]:          0
gffp-pkt[0x7040074]:          0
gffp-pkt[0x7040078]:          0
gffp-pkt[0x704007c]:          0
gffp-pkt[0x7040080]:          0
gffp-pkt[0x7040084]:          0
gffp-pkt[0x7040088]:          0
gffp-pkt[0x704008c]:          0
gffp-pkt[0x7040090]:          0
gffp-pkt[0x7040094]:          0
gffp-pkt[0x7040098]:          0
gffp-pkt[0x704009c]:          0
gffp-pkt[0x70400a0]:          0
gffp-pkt[0x70400a4]:          0
gffp-pkt[0x70400a8]:          0
gffp-pkt[0x70400ac]:          0
gffp-pkt[0x70400b0]:          0
gffp-pkt[0x70400b4]:          0
gffp-pkt[0x70400b8]:          0
gffp-pkt[0x70400bc]:          0
gffp-pkt[0x70400c0]:          0
gffp-pkt[0x70400c4]:          0
gffp-pkt[0x70400c8]:          0
gffp-pkt[0x70400cc]:          0
gffp-pkt[0x70400d0]:          0
gffp-pkt[0x70400d4]:          0
gffp-pkt[0x70400d8]:          0
gffp-pkt[0x70400dc]:          0
gffp-pkt[0x70400e0]:          0
gffp-pkt[0x70400e4]:          0
gffp-pkt[0x70400e8]:          0
gffp-pkt[0x70400ec]:          0
gffp-pkt[0x70400f0]:          0
gffp-pkt[0x70400f4]:          0
gffp-pkt[0x70400f8]:          0
gffp-pkt[0x70400fc]:          0
```

9.1.40 MIRROR CONTROL SET COMMAND

The ‘mirctl set’ command is used to modify the port mirroring control register. The command takes either 3 or 4 arguments.

```
metro-sw> mirctl set <linkIdx> <mtp> <dst_map> <fmt>
```

<linkIdx> is the link index for which the control register is set.

<mtp> mirror-to-port. This specifies the local port to which the mirrored packets are being sent.

<dst_map> Destination bitmap specifying which ports are mirrored on egress.

<fmt> Preserve format. When set, packets that are mirrored will be the same tagged/untagged format (at the mtp) as the ingress packet. When clear, the format of the mirrored packet is determined by the VLAN. This argument is optional. When omitted it is set implicitly.

Note: To make the settings effective, the configuration has to be saved and the switch has to be reset.

Usage example:

```
metro-sw> mirctl set 3 0 0xff0
```

The above command will set the mirror control register for port 3. All incoming packets on port 3 with destination port 11-4 will be mirrored to port 0. The preserve format bit is set implicitly.

9.1.41 MIRROR CONTROL SHOW COMMAND

The ‘mirctl show’ command is used to display the modified values of the port mirror control register.

```
metro-sw> mirctl show <linkIdx>
```

Usage example:

```
metro-sw> mirctl show 3
Mirror Control Register
link    reg             fmt     mtp     DSTmap
0x03    0x00140ff0      0x1     0x00    1111 1111 0000 [0xff0]

metro-sw> mirctl show all
Mirror Control Register
link    reg             fmt     mtp     DSTmap
0x01    0x0014000c      0x1     0x00    0000 0000 1100 [0x00c]
0x03    0x00140ff0      0x1     0x00    1111 1111 0000 [0xff0]
0x0b    0x000487ff      0x0     0x02    0111 1111 1111 [0x7ff]
```

reg: 32 bit register value.

fmt: Preserve format flag.

mtp: Mirror-to-port.

DSTmap: Destination bitmap binary value [hex value]

9.1.42 MIRROR CONTROL REGS COMMAND

The ‘mirctl regs’ command is used to display the actual values of the port mirror control registers currently used by the switch. It has the same output format as the ‘mirctl show’ command.

```
metro-sw> mirctl regs <linkIdx>
```

Usage example:

```
metro-sw> mirctl regs all
Mirror Control Register
link    reg             fmt     mtp     DSTmap

0x00    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x01    0x0014000c      0x1     0x00    0000 0000 1100 [0x00c]
0x02    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x03    0x00140ff0      0x1     0x00    1111 1111 0000 [0xff0]
0x04    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x05    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x06    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x07    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x08    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x09    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x0a    0x00000000      0x0     0x00    0000 0000 0000 [0x000]
0x0b    0x000487ff      0x0     0x02    0111 1111 1111 [0x7ff]
```

9.1.43 MIRROR CONTROL CLEAR COMMAND

The ‘mirctl clr’ command clears the mirror control register for the specified link.

```
metro-sw> mirctl clr <linkIdx>
```

9.1.44 EGRESS MASK (BLOCK MASK) SET COMMAND

The ‘egrmask set’ command is used to set the per-port egress mask (blocking mask). Each port allows a blocking of packets on egress, specified by the egress mask. A ‘0’ indicates no blocking, a ‘1’ indicates blocking of packets.

```
metro-sw> egrmask set <linkIdx> <egrMask>
```

<linkIdx> Link index for which the egress mask is to be set.

<egrMask> 12 bit value of egress mask, each bit representing a port. ‘0’ do not block, ‘1’ block (default value is ‘0’).

Usage example:

```
metro-sw> egrmask set 7 0xffe
```

Port 7 allows only packets to be forwarded which have destination port 0. All other incoming packets on port 7 are being blocked.

9.1.45 EGRESS MASK (BLOCK MASK) SHOW COMMAND

The ‘egrmask show’ command is used to display the value of the modified per-port egress mask.

```
metro-sw> egrmask show <linkIdx>
```

Usage example:

```
metro-sw> egrmask show 7
link             egress mask
0x07            1111 1111 1110 [0xffe]
```

9.1.44 EGRESS MASK (BLOCK MASK) REGS COMMAND

The ‘egrmask regs’ command is used to display the value of the actual per-port egress mask currently used by the switch.

```
metro-sw> egrmask regs <linkIdx>
```

Usage example:

```
metro-sw> egrmask regs 7
link             egress mask
0x07            1111 1111 1110 [0xffe]
```

9.1.45 EGRESS MASK (BLOCK MASK) CLEAR COMMAND

The ‘egrmask clr’ command clears the egress mask for the specified link.

```
metro-sw> egrmask clr <linkIdx>
```

9.2 TEST METHODS

There are several ways to test the Metro-Switch using methods to stimulate network traffic through the switch. This section provides ***suggestions*** on how you may test and verify your installation. Before you can proceed with any of the suggested tests, you must have previously configured your TCP/IP protocols and interfaces on your computers or external test equipment. Please refer to your online help and User Manuals for your particular system on instructions on setting up your TCP/IP environment.

- **ICMP Ping**. Ping is a simple verification method and it very useful for verification of cabling, switch and system configuration of the TCP/IP protocol software. You can also specify a “fast ping” and larger message sizes in order to more effectively test the protocols and switch capabilities.

- **FTP File Transfer**. FTP is also an available method for testing your installation. FTP allows you to test by transferring files of different sizes. Using large file transfers is a good way to test medium transfer rates through the switching system.

- **HTTP Web Browsing.** Web browsing is a good means of testing the protocols and switching through the system.

- **Telnet**. Using Telnet you can log into another system and invoke commands to send a receive data exercising interface and protocol functions in the switch. You can also test multiple connections within a single terminal window by repetitively Telnet’ing back and forth between multiple systems.

- **Windows Explorer**. In a Windows environment, you can also use the Windows explorer to access files on other systems within your Workgroup or Domain.

- **Blaster / blastee** can be used to perform TCP throughput tests between VxWorks, Linux and Windows platforms through the Metro-Switch. These test programs are included on the OEM developer CD.

- **Internal loopback**: Switch ports may be put into internal loopback to perform testing functions.

10. ON BOARD FIRMWARE

10.1 INTRODUCTION

The Metro-Switch has an onboard 16-bit RISC microcontroller and uses the Motorola DSP56F826 hybrid DSP/microcontroller for onboard management functions. The DSP56F826 is useful for a local switch management processor because it contains more features than found in conventional microcontrollers. It has a 16-bit bus interface unit suitable for external devices including SRAM and also has a JTAG interface suitable for use with a JTAG debugger.

The microcontroller switch management processor contains the following features:

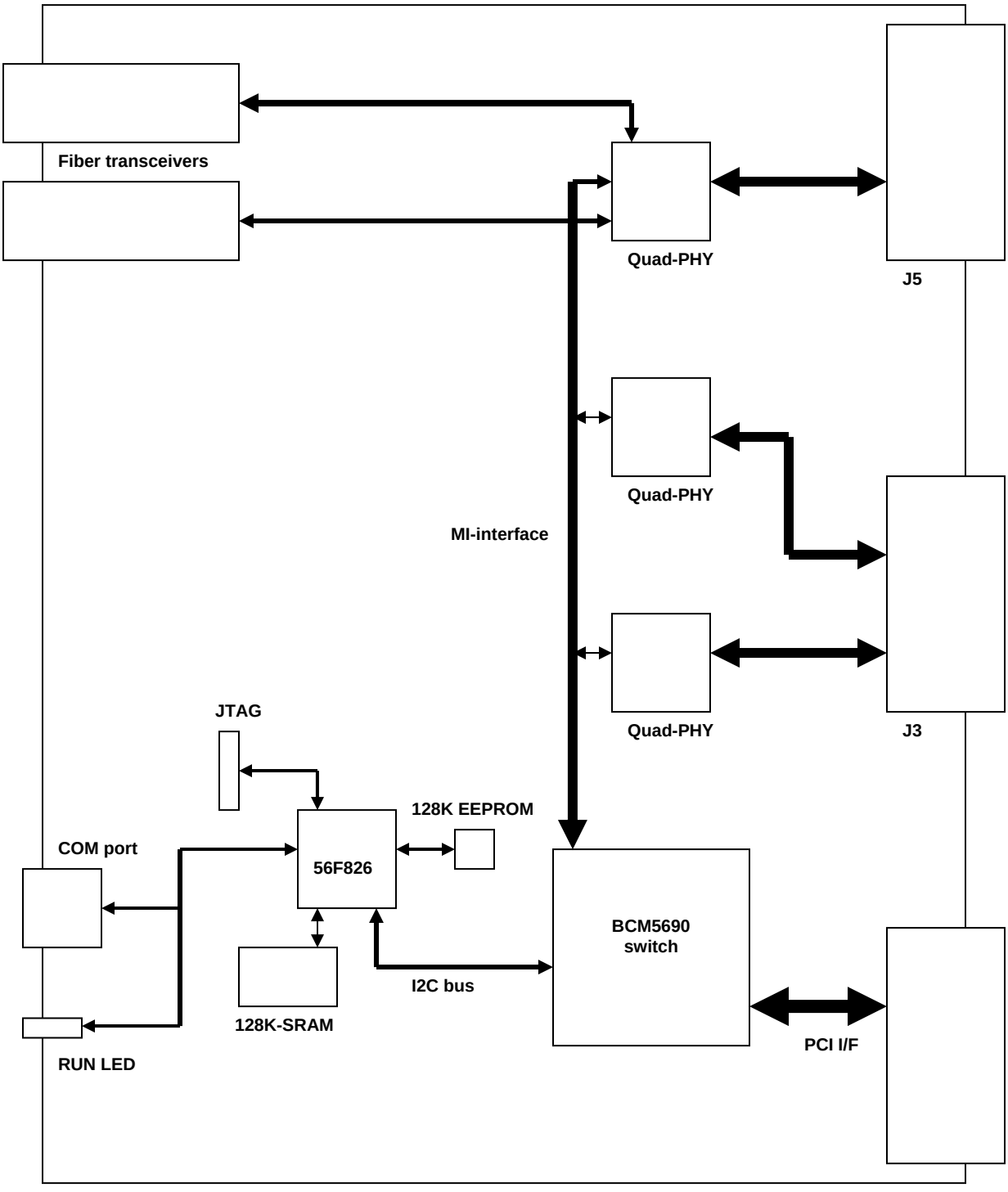
- **80 MHZ operation**
- **40 MIPS**
- **16-bit external bus interface unit**
- **I2C communications to BCM5690 switch at 100KHZ (12 Kbytes/sec)**
- **128K words (256KB) of SRAM for code and data**
- **16K bytes of EEPROM for switch configuration**
- **JTAG port connected to 14-pin onboard header**
- **3 general purpose LEDS (used for run-state)**
- **RS232 serial port connected to onboard RJ-11**

The microcontroller interfaces to the Broadcom BCM5690 switch via the I2C interface. This interface runs at 100KHZ and provides a register read/write access with a bandwidth of 100Kb (12KB/sec).

The microcontroller also controls each of the four Broadcom BCM5464SR quad-port transceivers by accessing them via the “MI” interface by going indirectly through the switch.

The microcontroller interfaces to the 128 KB general purpose non-volatile storage EEPROM via the I2C interface and interfaces to the 128KB SRAM via the 16-bit external bus interface. There is also a serial RS232 com port for the CLI command line interface and 3-LEDS are also accessible and controlled from the microcontroller.

10.2 BLOCK DIAGRAM



Switch data path block diagram

10.3 FIRMWARE DEVELOPMENT ENVIRONMENT

The development environment uses the Metrowerks IDE 5.1 in conjunction with the Motorola Embedded SDK version 3.0. The Metrowerks development environment provides a complete project-based facility and includes everything needed to develop, compile, link, download and debug firmware for the 56F826 via the JTAG port with “one-click” operation from compile to JTAG debug.

Metrowerks CodeWarrior IDE for Motorola 56800/56800E version 5.1 (current version 5.2.1177)

- Develop code using IDE or other editor
- Compile code using project facility
- Link code using project facility
- Download code from IDE onto board using JTAG interface
- Begin execution and debug using integrated debugger.

11. SPECIFICATIONS

Distance (copper): Recommended maximum distance is 328 feet (100 meters).

Backplane I/O: 12-ports routed to “J3” and “J5” Compact PCI connectors as per PICMG 2.16 specification (RJ3 and RJ5 on rear I/O backplane).

NOTE: All J3 and J5 (RJ3, RJ5) pins must be fully PICMG 2.16 compliant or Passive or electrical short circuit could occur.

Connectors (Rear I/O module):
10/100/1000 Base T connector type: RJ-45 (CAT5 specification)

Cable type (copper): CAT5, CAT5e or CAT6

Connector (Fiber, 1000 Base SX multimode):
LC type connector for 850nm VCSEL laser over 50 and 62.5 micron fiber

Connector (Fiber, 1000 Base LX singlemode):
LC type connector for 1310nm VCSEL laser over 9/125 micron fiber

Port Status Indicators: Link, Transmit, Receive and Signal Quality

Microcontroller Status Indicators: RUN, LINK STATUS, ERROR indicators

Bus Interface: PCI v2.2 bus master, 32-bit, 33/66 MHz

Dimensions: **Compact PCI 6U,** 9.187 (H) X 6.300 (W)

PCI Power supply voltage: 5V power rail from CPCI J1 connector

PCI signaling voltage: 5V and 3.3V (5V tolerant)

Performance Throughput: 24 Gbps (full-duplex) sustained

Maximum frame rate: Up to 32 million frames per second total aggregate switching rate (non-blocking)

Other: Jumbo packets, 802.3x full duplex flow control with automatic pause and priority with multiple priority queues

Voltage: Uses single 5V from CPCI connector “J1” Power Rail

Power: Uses single 5V supply from Compact PCI connector,
(onboard 3.3V, 2.5V and 1.2V regulators)

Power consumption: 18W total board power

Conformal Coating:
Conformal coating is available as custom order option and typically based on customer’s requirements and reference specifications.

MTBF: 200,000 hours

SAFETY: PCB manufactured with UL flammability rating of 94V-0

11.1 ENVIRONMENTAL SPECIFICATIONS

Rugged Class	Grade	Operating Temp.	Vibration	Shock	Humidity	Conformal Coated
C1	Commercial	0ºC to +65ºC 250 linear ft/minute air flow Storage: -55ºC	5Hz-2000Hz at 2g, 0.38mm peak displacement (operating) 5Hz-2000Hz at 5g, 0.76mm peak displacement (non-operating	20g, 11ms, ½ sine (operating); 30g, 11ms, ½ sine (non-operating)	Operating: Up to 90% Non-Condensing	No
R1	Rugged, Forced Air	-20ºC to +75ºC 500 linear ft/minute air flow Storage: -55ºC	5Hz-2000Hz at 2g, 0.38mm peak displacement (operating) 5Hz-2000Hz at 5g, 0.76mm peak displacement (non-operating	20g, 11ms, ½ sine (operating); 30g, 11ms, ½ sine (non-operating)	Operating: Up to 95% Non-Condensing	Yes
R2	Rugged, Forced Air	-40ºC to +85ºC 500 linear ft/minute air flow Storage: -55ºC	5Hz-2000Hz at 2g, 0.38mm peak displacement (operating) 5Hz-2000Hz at 5g, 0.76mm peak displacement (non-operating	20g, 11ms, ½ sine (operating); 30g, 11ms, ½ sine (non-operating)	Operating: Up to 95% Non-Condensing	Yes

Environmental Standards Compliance (pending):

- FCC Part 15, Class B
- EN 55022; 1998 Class B
- EN 50082-1
- CE Mark

11.2 MECHANICAL SPECIFICATIONS

- 6U form-factor: 9.187 inches x 6.3 inches (233mm x 160mm)
- Single-slot-width: 0.8inches (20.3mm)
- Connectors: IEC-1076-4-101 type for J1-J5

11.3 INDUSTRY STANDARDS COMPLIANCE

- PICMG 2.0 R3.0 Compact PCI compliant (6U form factor)
- PICMG specifications: 2.16 R1.0 Packet Switching Backplane (PSB) complaint fabric board (occupies one fabric slot per board, up to two per chassis)
- PICMG 2.9 R1.0 IPMI support
- PICMG 2.1 R2.0 hot-swap (basic hot-swap hardware connection layer)
- IEEE specifications: 802.3-2002, 802,1D, 802.1Q, 802.3x, 802.3z
- Spanning tree: IEEE 802.1D (STP), 802.1s (MSTP), 802.1w (RSTP)
- RMON, SNMP and Ethernet MIB statistics, RFCs 1213, 1493, 1573, 1757, 2358 and IEEE 802.3 Ethernet MIB
- 12 ports 1000 Base T routed to backplane
- Two onboard 850nm multimode fiber LC connectors (1000 Base SX) via front panel interface
- Optional support for two 1310nm singlemode fiber LC connectors (1000 Base LX)
- 10-Gigabit Ethernet fiber uplink (extended model)

11.4 SNMP SUPPORT MIB COUNTERS

The switch provides many counters to support system management capabilities. Each port has over 70 counters. Counters are provided to support the following SNMP MIBs:

- RMON Statistics Group (RFC 1757)
- SNMP Interface Group (RFC 1213 and 1573)
- Ethernet-like MIB (RFC 2358)
- Bridge MIB (RFC 1493)

- Ethernet MIB (RFC 802.3)

11.5 PERFORMANCE AND CAPABILITIES

Full Duplex:
Support for 10/100/1000 Base T on all copper models, auto-negotiating
1000 Base X on fiber ports, auto negotiating

Virtual Network: Virtual LAN (VLAN) tag support

Line speed: multi-layer switching on all 12-ports

Performance category: 32 million packets-per-second/24Gb line rate switching

L2 table: 16384-entry ARL MAC address table, automatic learning

L3 Table: Integrated 4096-entry IPv4 host table

Port trunking and mirroring: (32 trunk groups with link aggregation)

Memory: Integrated 1MB packet memory

Protocol support:
Rapid spanning tree protocol support
VLAN 802.1D, 802.1Q support
(4096 VLANS, 8 Classes of service per port)
IP multicasting

Traffic control:
Packet classification and L2-L7 filtering
Traffic metering/shaping

Flow control: Advanced flow-control and head-of-line blocking prevention, packet aging, storm control

Filtering: Advanced fast filter processor and rules-based matching

Jumbo Frames: Supports jumbo frames to 9K

11.6 MANAGEMENT FEATURES

- Can be operated in simple (lightly) managed or unmanaged mode
- Simple managed switch operation with serial port CLI console management interface
- External management mode via PCI bus interface and external processor
- Onboard management provided by microprocessor and firmware
- Management functions for all onboard devices including Broadcom BCM5690 switch and BCM5464SR transceivers
- Configuration, status, statistics, diagnostics and healthy status

11.7 HARDWARE SPECIFICATIONS

- Broadcom BCM5690/5695 12-port non-blocking multi-layer switch fabric network processor
- Broadcom BCL6464SR 4-port gigabit transceivers with Serdes
- Motorola DSP56F826 management processor, 80 MHZ
- PIGMG 2.1 R1.0 hot-swap support for hardware connection layer
- IPMI management interface
- 32-bit, 66 MHZ PCI bus interface
- Serial I2C EEPROM for configuration parameters, up to 128K X 8
- 2X onboard 850nm multimode fiber LC connectors for 1000 base SX
- 2X 1310nm singlemode fiber LC connectors for 1000 base LX (optional)
- 4 multi-function LEDS per port, 3 CPU controlled multi-function LEDS via front panel
- 12-port rear I/O module with ganged CAT5 RJ-45 connectors
- RJ-11 serial port console connector via front panel

12. WARRANTEE AND SUPPORT INFO

Technical Support and Warranty:

Telephone technical support (8AM to 6PM, MST), 24-hour support via web email
1 year product warranty on controller hardware (subject to Standard Warrantee Policy and Purchase Terms and Conditions).

Contacting Us

You may contact DSS Networks in one of several ways: via the Web, e-mail, fax or telephone.

Sales and Technical Support

Send all technical support queries to support@dssnetworks.com or visit the DSS Networks website at www.dssnetworks.com. The website contains product, technical and sales information.

Sales and Technical Support-Worldwide
+1.949.716.9051

Sales and Technical Support-Fax
+1.949.716.9052

Main Corporate Telephone Numbers
949-716-9051