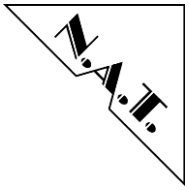
NAT-MCH

User's Manual

Version 1.32



NAT-MCH has been designed by:

N.A.T. GmbH
Konrad-Zuse-Platz 9
53227 Bonn-Oberkassel

Phone: +49 / 228 / 96 58 64 0
Fax: +49 / 228 / 96 58 64 10

Internet: http://www.nateurope.com

Disclaimer

The following documentation, compiled by N.A.T. GmbH (henceforth called N.A.T.), represents the current status of the product's development. The documentation is updated on a regular basis. Any changes which might ensue, including those necessitated by updated specifications, are considered in the latest version of this documentation. N.A.T. is under no obligation to notify any person, organization, or institution of such changes or to make these changes public in any other way.

We must caution you, that this publication could include technical inaccuracies or typographical errors.

N.A.T. offers no warranty, either expressed or implied, for the contents of this documentation or for the product described therein, including but not limited to the warranties of merchantability or the fitness of the product for any specific purpose.

In no event, will N.A.T. be liable for any loss of data or for errors in data utilization or processing resulting from the use of this product or the documentation. In particular, N.A.T. will not be responsible for any direct or indirect damages (including lost profits, lost savings, delays or interruptions in the flow of business activities, including but not limited to, special, incidental, consequential, or other similar damages) arising out of the use of or inability to use this product or the associated documentation, even if N.A.T. or any authorized N.A.T. representative has been advised of the possibility of such damages.

The use of registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations (patent laws, trade mark laws, etc.) and therefore free for general use. In no case does N.A.T. guarantee that the information given in this documentation is free of such third-party rights.

This product (and the associated documentation) is governed by the N.A.T. General Conditions and Terms of Delivery and Payment.

Table of Contents

List of Figures

List of Tables

Glossary

AMC	Advanced Mezzanine Card
ATCA	Advanced Telecom Computing Architecture
BMC	Baseboard Management Controller
BT	Block Transfer
CM	Carrier Manager
CPU	Central Processing Unit
CU	Cooling Unit, Fan
EMMC	Enhanced Module Management Controller (MMC on CU or PU), via IPMB-0
FRU	Field Replaceable Unit, hotswap capable resource
HPI	Hardware Platform Interface, SW management interface defined by SAF
I2C	Inter Integrated Circuit, 2 wire serial bus
IPM	Intelligent Platform Management
IPMB	IPM Bus, I2C type
IPMB-0	dual-redundant A/B local IPMB
IPMB-L	non-redundant local IPMB
IPMC	IPM Controller, e.g. MCH CPU
IPMI	IPM Interface
KCS	Keyboard Controller Style
LAN	Local Area Network
LED ID	Light Emitting Diode Identifier
LUN	Logical Unit Number
LVDS	Low Voltage Differential Signal
MCH	uTCA Carrier Hub
MCMC	uTCA Carrier Manager Controller (MMC on MCH)
MMC	(AMC) Module Management Controller, interfaced to carrier via IPMB-L
MTCA	Micro Telecommunications Computing Architecture (= uTCA)
MTCM	uTCA Carrier Manager
NetFn	Network Function, functional class of message
NMCH	N.A.T. MCH
OEM	Original Equipment Manufacturer
OpenHPI	specific HPI implementation, http://www.openhpi.org/
PEF	Platform Event Filtering
PET	Platform Event Trap
PM	Power Module (= PU)
POH	Power On Hours
PU	Power Unit (= PM)
RCS	Remote Console Software, SW running on a remote system
RDR	Resource Data Record, logical representation of physical entity
RMCP	Remote Management Control Protocol, UDP based, IPMI over LAN
RPT	Resource Presence Table, data base of logical resources
SAF	Service Availability Forum, http://www.saforum.org/
SDR	Sensor Data Record, sensor description
SEL	System Event Log
SM	Shelf Manager

SMI	System Management Interrupt
SMIC	Server Management Interface Chip, type of interface to an IPMI BMC
SMS	System Management Software, SW running on BMC
SSID	System Software Identifier
SSOID	System Sensor Owner Identifier
SW	Software
TCA	Telecom Computing Architecture
UDP	User Datagram Protocol
uTCA	micro TCA (= MTCA)

1 Board Specification

MCH-Module	MicroTCA MCH-Module, single width, full size; MCH M4: double width, full size	
Processor	MCH Gen2: ColdFire MCF5470, 200 MHz MCH Gen 3: MCH-LC: Coldfire MCF54452, 266 MHz	
Front-I/O	2 RJ45 connectors, 1 (2) SMA and 1 Mini-USB connector	
Main Memory	32/64 MByte SDRAM	
Flash PROM	16/32/64 MByte Flash PROM, on board programmable	
Operating System	OK-1	
Power consumption	Base Module: 8.5W typ. Mezzanine Modules for tongue 2/3/4 adding: Clock: 8W PCIe: 6W + 1W per active connection XAUI: 29W SRIO: 15W	
Environmental conditions	Temperature (operating): Temperature (storage): Humidity:	0°C to +65°C with forced cooling -40°C to +85°C 10 % to 90 % rh non-condensing
Standards compliance	PICMG AMC.0 Rev. 2.0 PICMG AMC.2 Rev. 1.0 PICMG SFP.0 Rev. 1.0 (System Fabric Plane Format) IPMI Specification V1.5 Rev. 1.0 PICMG µTCA.0 Rev. 1.0	

Table 1-1 : NAT-MCH Technical Data

1.1 NAT-MCH Features

1.1.1 CPU and Memory

MCH Gen 2: Freescale ColdFire 5470 @ 200MHz
MCH Gen 3, LC: Freescale ColdFire 54452 @ 266MHz
DRAM: 64MB
FLASH: 32MB

1.1.2 IPMI and Management

The MCH supports Management and IPMI Interfaces for:

- 12 AMCs,
- 2 cooling units
- 1-4 power units

1.1.3 Supported Fabrics and Compliance

Fabric A: **Gigabit Ethernet Option**
non-blocking, low-latency Layer 2 Gigabit Ethernet switch
Support for 12 AMCs and 1 GbE Uplink Port
PICMG AMC.2 R1.0
PICMG SFP.1 R1.0

Fabric D-G: **PCI Express Option**
12 AMCs, x1-x4 each
PICMG AMC.1 R1.0

Serial Rapid I/O Option
12 AMCs + dual face plate uplink
PICMG AMC.4

XAUI (10GbE) Option
12 AMCs + dual face plate uplink
PICMG AMC.2

1.1.4 Clock Distribution

- Telecom: Stratum 3 PLL with reference from either 1 of the 12 AMCs or external clock via front panel
- PCIe: Spread Spectrum Clock (100MHz mean) or oscillator (100MHz fixed)

1.1.5 External Clock Reference Input

- External Clock input for Telecom clocking module via front panel SMA1/SMA2
- Supported input frequencies: 1pps, 2kHz, 8kHz, 1.544MHz, 2.048MHz, 8.192MHz, 16.384MHZ, 19.44MHZ

- Input Range: 300mV-5V

1.1.6 LEDs

- Standard LEDs according to AMC.0 specification
- Bi-color indicator LEDs for status indication of each of the AMC modules, 2 cooling units and 2 power modules.

2 Overview: NAT-MCH Product Family

The NAT-MCH is a MicroTCA Carrier Hub in the form factor of a single width, full size Advanced Mezzanine Card (AMC). It provides the central management and data switching functionality for all MicroTCA systems. The NAT-MCH comprises of a base module and numerous optional daughter cards which can be mounted on the base module. The NAT-MCH is MicroTCA.0 R1.0 compliant and delivers switching and hub functionality for the various system fabrics as defined in the AMC.x standard series, i.e. Gigabit Ethernet (GbE), PCI-Express (PCIe), Serial Rapid I/O (SRIO) or XAUI (10Gigabit Ethernet). The NAT-MCH can also provide a centralized clock distribution to all AMCs in the system. The following block diagram gives an overview about the main MCH building blocks.

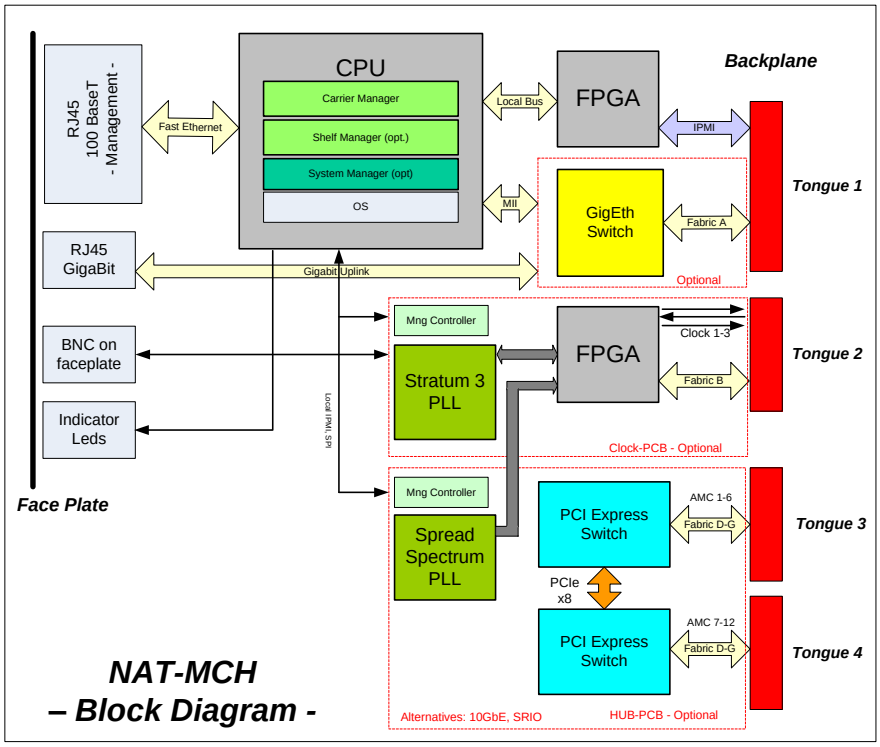
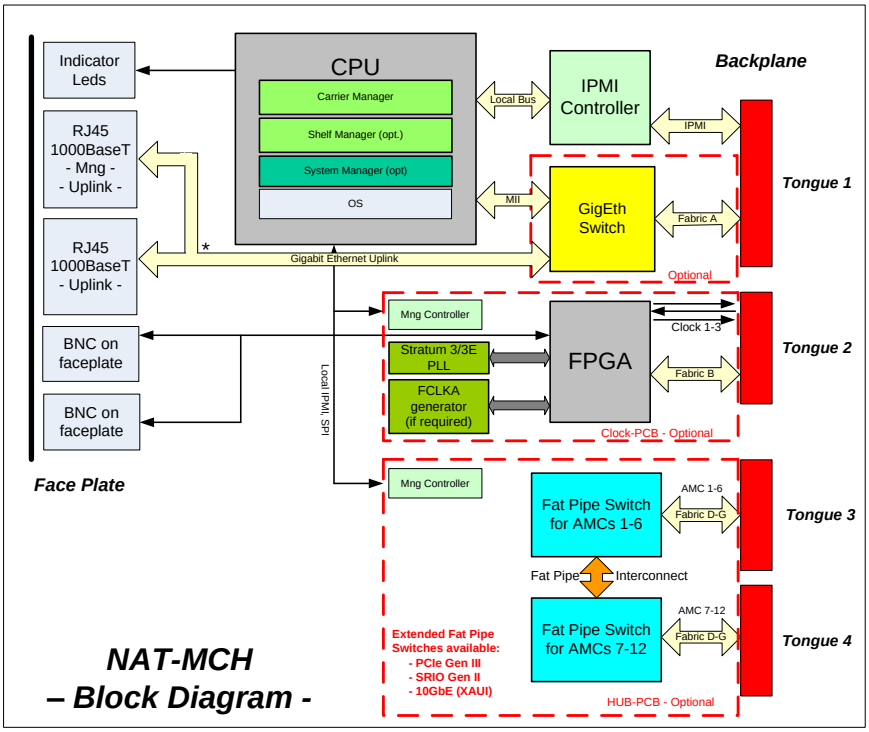


Figure 1 : NAT-MCH Gen 2 - Block Diagram with PCIe Mezzanine Option shown



* 2nd Ethernet port only available if MCH is not in Managed Ethernet Mode

Figure 2 : NAT-MCH Gen 3 - Block Diagram with PCIe Mezzanine Option shown

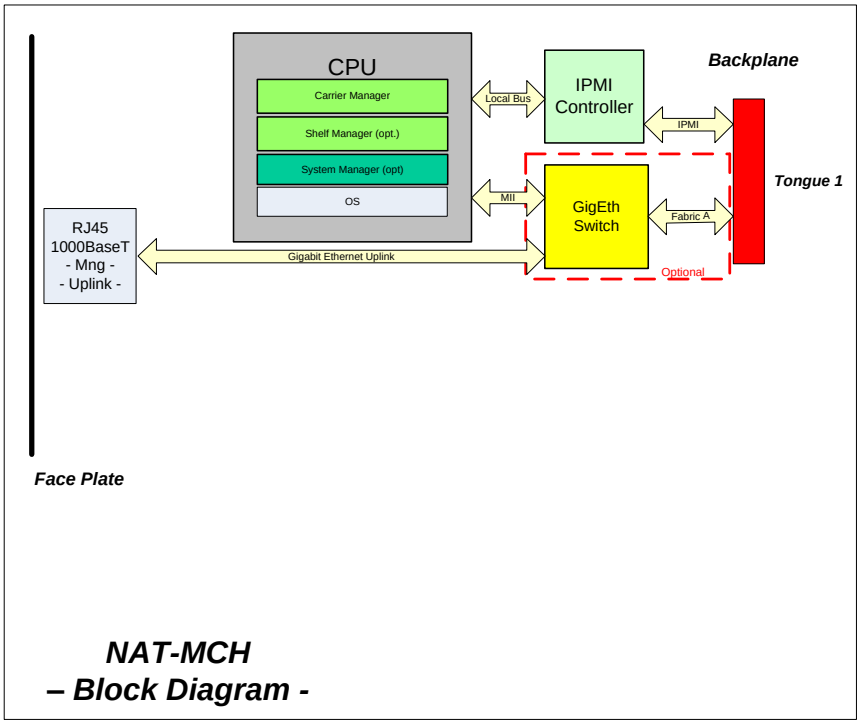


Figure 3 : NAT-MCH Gen LC - Block Diagram with PCIe Mezzanine Option shown

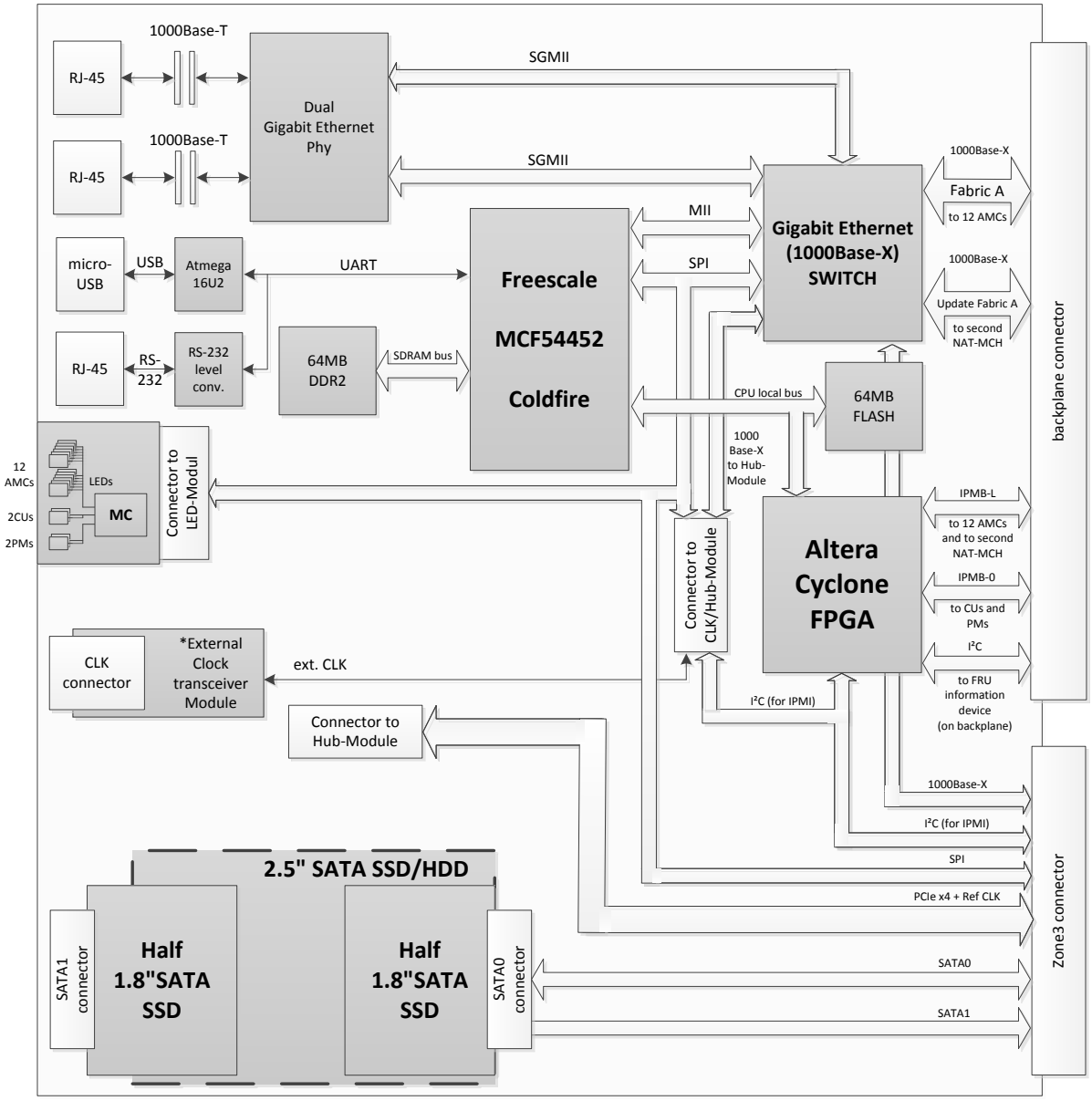


Figure 4 : NAT-MCH M4 - Block Diagram with SSD Option shown

2.1 NAT-MCH Variants

The following table lists the available variants and its features for the NAT-MCH:

NAT-MCH	LC (low cost)	Gen2	Gen3	M4
Base Functionality:	●	●	●	●
Management of up to 12 AMCs 2 CUs, 4 PMs	●	●	●	●
Onboard Shelf Manager	●	●	●	●
Console Port	USB	RS232	USB	USB
RMCP (GbE) based management interface	●	●	●	●
GbE Uplink Ports	1	1	2	2
Port Trunking with second GbE Uplink Port			●	●
Power fail safe SDR Repository			●	●
Bicolour LEDs for status indication of AMCs, PMs and CUs		●	●	●
Clock and Fat Pipe mezzanines		●	●	●
Real Time Clock		●	●	●
JTAG/JSM support			●	●
Faceplate:	Full/Mid/ Compact	Full	Full/Mid	Full
Clock Module Mezzanine:		●	●	●
AMC clocks supported		1,2,3 F_CLK	1,2,3 F_CLK	1,2,3 F_CLK
PLL Accuracy		Stratum 3	Stratum 3 + Stratum 3E (Option)	
Face plate reference clock		1 (input)	2 (input/output)	2 (input/output)
Clock Input Amplifier		fixed	modular	modular
Fabric Clock Support (PCIe)		HCSL	HCSL	HCSL
Spread Spectrum support (F_CLK)		●	●	●
Fat Pipe Mezzanines		●	●	●
Variants		PCIe, XAUI, SRIO	PCIe, XAUI, SRIO	PCIe
Uplink ports at face plate			2 XAUI,SRIO	2 GbE
PCIe Support		Gen 1	Gen1, Gen2, Gen3	Gen1, Gen2, Gen3
GbE Support		Type 1,5,6	Type 1,5,6	Type 1,5,6
Management Software				
External User Interface	RMCP	RMCP	RMCP	RMCP
NATview – Graphical User Interface	●	●	●	●
(open)HPI	●	●	●	●
IPMI Tool	●	●	●	●

Table 2-1: MCH Variants and Features

2.2 NAT-MCH Gen2: Connectors and Indicators at the Face Plate

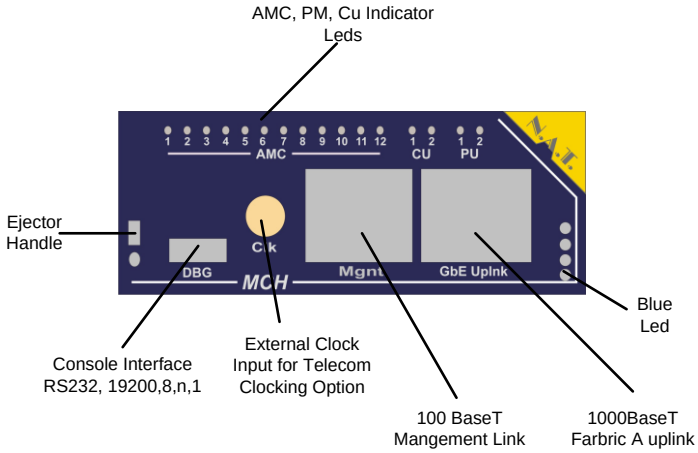


Figure 5 : NAT-MCH Gen 2 Front Panel

2.2.1 LED Indicators

The NAT-MCH is equipped with two sets of indicator LEDs:

- 4 indicator LEDs according to AMC.0 specification
- 16 indicator LEDs displaying the status of AMC modules, Cooling Units and Power Modules

The four AMC.0 conformant LEDs are assigned to the following functions:

- Blue LED: Blue LED function according to AMC.0
- Red LED: severe fault
- Green LED: MCH has taken over role of primary MCH
- Yellow LED: MCH is redundant

The 16 bi-color LEDs at the upper border of the MCH give an immediate visual feedback of the status of the corresponding module. Their functions are:

- green: AMC, CU or PM module fully inserted and operational
- green blinking – activation /de-activation under progress
- red – module faulty or did not progress into operational state, communication fault
- red blinking – PM error, e.g. power supply faulty

2.2.2 SMA Connector – External Clock Reference

This input can be used to provide the NAT-MCH with a system wide clock, which can be distributed by the Telecom Clocking Module to any AMC slot in the system. For technical data of this input please refer to the *NAT-MCH Features* section.

2.2.3 100 BaseT interface – primary Management Interface Port

The 100BaseT interface provides an autonegotiation 10/100 MBit link to the onboard CPU. This is the default management port and can be used by any external Shelf or System Manager to control the operation of the NAT-MCH and the system. The onboard CPU supports TCP/IP and RMCP accesses.
Note: The management port can be switched to the GbE uplink port by a configuration option (see chapter *Configuration* below).

2.2.4 GigaBit Ethernet (GbE) Uplink Port

The GbE Uplink Port provides a direct access to the onboard GbE Switch for Fabric A. Thus the Fabric connections can be extended to other shelfs or systems if required.
Optionally the GbE Uplink Port can take over the management port functionality (see above).

2.2.5 Console (Dbg) Port

The console port provides an interface to the Command Line Interface (**CLI**) of the onboard CPU. The console interface can be used to set the operational and configuration parameters of the NAT-MCH. Once the IP configuration has been set, the console interface can be switched to a Telnet or SSH session by connecting via Telnet/SSH. In case a password had been configured for a Telnet session a check of this password is done when starting the Telnet session. For SSH sessions this password check is mandatory. For details regarding configuration of a Telnet/SSH password, please refer to chapter 8.
Important Note: The console port has changed from Gen2 MCH to Gen3 MCH from a RS232 type interface to an USB type interface. To prevent any mismatch the connector type has been changed too, so that a false connection can be excluded as far as possible.

2.2.5.1 Console Port of MCH V1.x and V2.x (Gen1 and Gen2) – RS232

The MCHs of generation 1 and 2 have a RS232 based console port interface.

The default parameters of the console port are: **19200, 8, N, 1**

The default baud rate can be changed within the MCH configuration settings (see Chap. 5).

Important Note: Despite the connector type may indicate that the console port of the NAT-MCH is an USB port it is realized as a RS232 interface. Never connect this port to the USB port of a computer or to a hub. Unpredictable damage might be the result.

2.3 NAT-MCH Gen3: Connectors and Indicators at the Face Plate

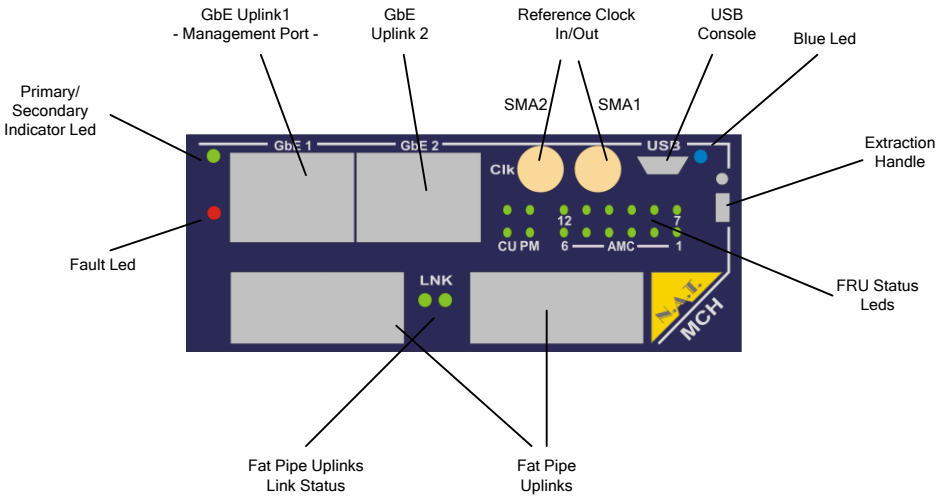


Figure 6 : NAT-MCH Gen 3 Front Panel with SRIO/XAUI uplink (optional)

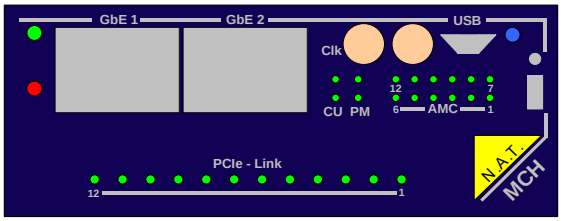


Figure 7 : NAT-MCH Gen 3 Front Panel with PCIe-option

2.3.1 LED Indicators

The NAT-MCH is equipped with three sets of indicator LEDs:

- 3 indicator LEDs according to AMC.0 specification
- 16 indicator LEDs displaying the status of AMC modules, Cooling Units and Power Modules
- 2 link indication LEDs for Fatpipe Link Status (SRIO-/XAUI-option only) **OR** 12 link indication LEDs for PCIe link status (PCIe Gen3 only)

The three AMC.0 conformant LEDs are assigned to the following functions:

- Blue LED: Blue LED function according to AMC.0
- Red LED: severe fault
- Bicolor Led Green/Yellow:
  - Green: MCH is primary management controller
  - Yellow LED: MCH is redundant/standby

The 16 bi-color LEDs give an immediate visual feedback of the status of the corresponding FRU device. Their functions are:

- green: AMC, CU or PM module fully inserted and operational
- green blinking – activation /de-activation under progress
- red – module faulty or did not progress into operational state, communication fault
- red blinking – PM error, e.g. power supply faulty

2.3.2 SMA Connectors – External Clock Reference

This input/output can be used to feed an external reference clock into the NAT-MCH or provide a reference clock for other systems. The input clock can be distributed by the Telecom Clocking Module to any AMC slot in the system. The output can provide a clock sourced from any of the AMCs or a local clock generated by the onboard PLL.
For technical data of this input/output please refer to the *NAT-MCH Features* section.

2.3.3 Dual GbE Uplink Port

The NAT-MCH has two 10/100/1000 BaseT (autonegotiation) uplink ports which are connected to the internal GbE switch circuit.
The ports can be joined together in link aggregation mode to double the uplink throughput performance. In alternative, they can be used as separate ports of the Ethernet Switch, e.g. with using VLAN functionality to divide up traffic paths.
If the MCH operates in Managed Ethernet Mode (Spanning Tree) only one front Ethernet interface is available as the switch connection to the other front interface is used internally.

2.3.4 Management Interface Port

One of the GbE uplink ports is used as the management interface for external hosts to communicate with the onboard Shelf/Carrier manager by RMCP. By default the management port is assigned to GbE uplink port 1.
The management port can be used by any external Shelf or System Manager to control the operation of the NAT-MCH and the system. The onboard CPU supports TCP/IP and RMCP accesses.

2.3.5 Console Port – USB / Telnet / SSH

The console port provides an interface to the Command Line Interface (**CLI**) of the onboard CPU. The console interface can be used to set the operational and configuration parameters of the NAT-MCH. Once the IP configuration has been set, the console interface can be switched to a Telnet or SSH session by connecting via Telnet/SSH. In case a password had been configured for a Telnet session a check of this password is done when starting the Telnet session. For SSH sessions this password check is mandatory. For details regarding configuration of a Telnet/SSH password, please refer to chapter 8.

2.3.5.1 Console Port of MCH V3.x (Gen3) - USB

The NAT-MCH of generation 3 (V3.x) uses a USB interface as console port. The console port provides a USB-CDC type of interface. Interoperability has been tested with the standard drivers included in Windows 2000, XP, Vista, Windows 7 and Linux.
When connecting first time to a windows machine a new device will be installed (NAT-MCH console). To successfully complete the installation a device information file must be provided to the system. N.AT. provides the required "***nat_mch.inf***" text file for download from its web site or ftp server.

Under Windows the new device can be accessed by standard terminal programs like "terraterm" (COMxx) port. The new device and the assigned COM port can be looked up in the Windows device manager.

Note: Do not start the terminal program before the USB link has been established, because the normal terminal programs needs an active link to connect to. To avoid the loss of log messages the MCH keeps a history buffer which can be recalled by the CLI command "history".

Under Linux the new device can be accessed via device descriptor "ttyXYZ" by standard terminal programs like "minicom". The name of the newly generated device descriptor is system dependent and needs to be looked up in the devices directory.

2.4 NAT-MCH-LC: Connectors and Indicators at the Face Plate

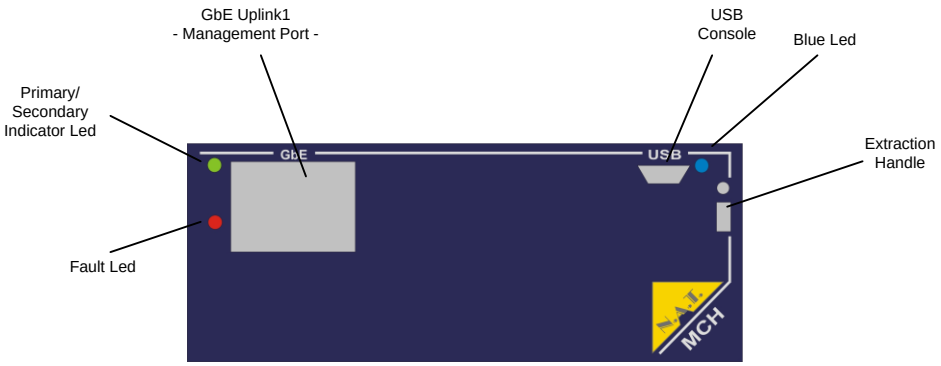


Figure 8 : NAT-MCH Gen LC Front Panel

2.4.1 LED Indicators

The NAT-MCH is equipped with 3 indicator LEDs according to AMC.0 specification

The three AMC.0 conformant LEDs are assigned to the following functions:

- Blue LED: Blue LED function according to AMC.0
- Red LED: severe fault
- Bicolor Led Green/Yellow:
  - Green: MCH is primary management controller
  - Yellow LED: MCH is redundant/standby

2.4.2 GbE Uplink Port

The NAT-MCH-LC has one 10/100/1000 BaseT (autonegotiation) uplink port. The port is connected to the onboard GbE switch. This is the designated management interface, too.

2.4.3 Management Interface Port

The GbE uplink port is used as the management interface port for external hosts to communicate with the onboard Shelf/Carrier manager by RMCP. The management port can be used by any external Shelf or System Manager to control the operation of the NAT-MCH and the system. The onboard CPU supports TCP/IP and RMCP accesses.

2.4.4 Console Port – USB / Telnet / SSH

The console port provides an interface to the Command Line Interface (**CLI**) of the onboard CPU. The console interface can be used to set the operational and configuration parameters of the NAT-MCH. Once the IP configuration has been set, the console interface can be switched to a Telnet or SSH session by connecting via Telnet/SSH.

For a detailed description, please refer to chapter 2.3 above.

2.5 NAT-MCH M4: Connectors and Indicators at the Face Plate

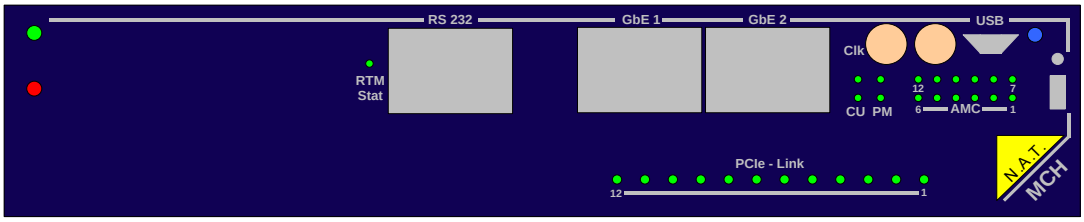


Figure 9 : NAT-MCH Gen 4 Front Panel

2.5.1 LED Indicators

The NAT-MCH is equipped with four sets of indicator LEDs:

- 3 indicator LEDs according to AMC.0 specification
- 1 indicator LED reflecting the RTM module's status
- 16 indicator LEDs displaying the status of AMC modules, Cooling Units and Power Modules
- 12 indicator LEDs for PCIe Link Status

The three AMC.0 conformant LEDs are assigned to the following functions:

- Blue LED: Blue LED function according to AMC.0
- Red LED: severe fault
- Bicolor Led Green/Yellow:
  - Green: MCH is primary management controller
  - Yellow LED: MCH is redundant/standby

The 16 bi-color LEDs give an immediate visual feedback of the status of the corresponding FRU device. Their functions are:

- green: AMC, CU or PM module fully inserted and operational
- green blinking – activation /de-activation under progress
- red – module faulty or did not progress into operational state, communication fault
- red blinking – PM error, e.g. power supply faulty

2.5.2 SMA Connectors – External Clock Reference

This input/output can be used to feed an external reference clock into the NAT-MCH or provide a reference clock for other systems. The input clock can be distributed by the Telecom Clocking Module to any AMC slot in the system. The output can provide a clock sourced from any of the AMCs or a local clock generated by the onboard PLL.
For technical data of this input/output please refer to the *NAT-MCH Features* section.

2.5.3 Dual GbE Uplink Port

The NAT-MCH has two 10/100/1000 BaseT (autonegotiation) uplink ports which are connected to the internal GbE switch circuit.
The ports can be joined together in link aggregation mode to double the uplink throughput performance. In alternative, they can be used as separate ports of the Ethernet Switch, e.g. with using VLAN functionality to divide up traffic paths.
If the MCH operates in Managed Ethernet Mode (Spanning Tree) only one front Ethernet interface is available as the switch connection to the other front interface is used internally.

2.5.4 Management Interface Port

One of the GbE uplink ports is used as the management interface for external hosts to communicate with the onboard Shelf/Carrier manager by RMCP. By default the management port is assigned to GbE uplink port 1.
The management port can be used by any external Shelf or System Manager to control the operation of the NAT-MCH and the system. The onboard CPU supports TCP/IP and RMCP accesses.

2.5.5 Console Port – USB / Telnet / SSH and RS232 via RJ45

The console port provides an interface to the Command Line Interface (**CLI**) of the onboard CPU. The console interface can be used to set the operational and configuration parameters of the NAT-MCH. Once the IP configuration has been set, the console interface can be switched to a Telnet or SSH session by connecting via Telnet/SSH. In case a password had been configured for a Telnet session a check of this password is done when starting the Telnet session. For SSH sessions this password check is mandatory. For details regarding configuration of a Telnet/SSH password, please refer to chapter 8.

2.5.5.1 Console Port of MCH M4 - USB

The NAT-MCH M4 uses a USB interface as console port. The console port provides a USB-CDC type of interface. Interoperability has been tested with the standard drivers included in Windows 2000, XP, Vista, Windows 7 and Linux.
When connecting first time to a windows machine a new device will be installed (NAT-MCH console). To successfully complete the installation a device information file must be provided to the system. N.AT. provides the required "***nat_mch.inf***" text file for download from its web site or ftp server.

Under Windows the new device can be accessed by standard terminal programs like "terraterm" (COMxx) port. The new device and the assigned COM port can be looked up in the Windows device manager.

Note: Do not start the terminal program before the USB link has been established, because the normal terminal programs needs an active link to connect to. To avoid the loss of log messages the MCH keeps a history buffer which can be recalled by the CLI command "history".

Under Linux the new device can be accessed via device descriptor "ttyXYZ" by standard terminal programs like "minicom". The name of the newly generated device descriptor is system dependent and needs to be looked up in the devices directory.

2.5.5.2 Console Port of MCH M4 – RS-232 via RJ45 Connector

The RJ45 Connector connects via a RS-232 level converter to the Coldfire UART to provide a serial console debug interface. It can be used as an alternative to the USB interface.

Please note: If the USB interface is connected to an USB host (e.g. a PC) the RS232 receive path will be disabled. That means if the serial console is connected to a terminal program the debug outputs of the MCH firmware will still show up but it will not be possible to interact.

3 Mezzanine Module Options

The NAT-MCH base board is a modular design and can be completed with additional functionalities by adding Mezzanine PCBs for the tongues 2-4.

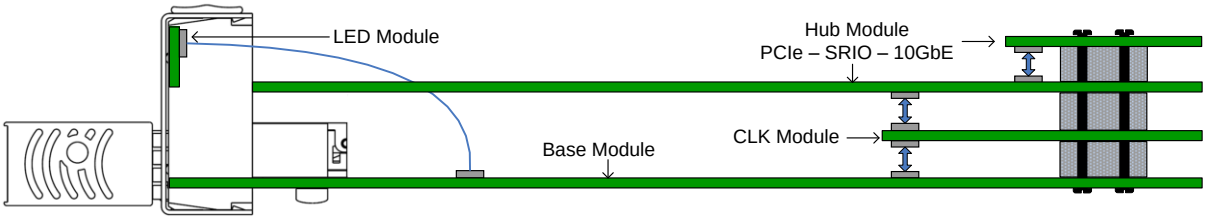


Figure 10 : NAT-MCH Mezzanine Options

3.1.1 Base Module

The NAT-MCH Base Module carries the main CPU, the IPMI controllers for up to 12 AMC modules, Power and Cooling Units and backplane devices and the front panel connectors. The Base module is available in two assembly options:

- Base 6 – for systems with up to 6 AMC modules
- Base 12 – for systems with up to 12 AMC modules

The Gen2 base module can optionally be equipped with a GbE Level 2 switch for Fabric A for up to 12 AMC modules, backplane backup link, and front panel uplink.

- Option: -GbE

The Gen3 base module is always equipped with the GbE Level 2 switch.

Note: The GbE option is mandatory for redundant operation of the Gen2 MCH.

3.1.2 Clock Modules

Available Options:

- -TC: Telecom Clocking
- -SSC: Spread Spectrum clocking for FCLK-A (only in combination with option –X24 or –X48)

3.1.2.1 Telecom Clocking Module

The clock module provides the telecom clocks CLK1 and CLK2 to the 12 AMC modules as well as the fabric clock FCLK-A.
The telecom clocks can be sourced from either of the AMC modules by means of CLK2 as an input to the MCH or from the front panel external clock reference input.

3.1.2.2 Spread Spectrum Clocking Module (SSC)

The Spread Spectrum Clocking module (SSC) distributes a 100 MHz clock to any of the AMC slots. The clock can be configured to be a standard 100 MHz clock or a 100MHz Spread Spectrum Clock (see Table 2). The configuration option '100 MHz spread spectrum clock' is only applicable when the PCIe Hub module isn't assembled as option LOSC. The clock is switched onto the individual AMC slots according to the E-Keying definitions acquired from the AMC module in the respective slot.

3.1.3 HUB Mezzanine Modules

Optionally the NAT-MCH can be equipped with mezzanine modules which provide the data switching functionalities for the fabrics D-G of the uTCA backplane.
The available and planned options are:

- -X24: PCIe Hub module for up to 6 AMC modules
- -X48: PCIe Hub module for up to 12 AMC modules
- -SRIO: Serial Rapid I/O module for up to 12 AMC modules
- -XAUI: (10 GbE) module for up to 12 AMC modules

For a detailed description of these modules and their technical data, please refer to the corresponding hardware manual.

3.1.3.1 PCI Express Switching Module (X24 and X48)

The PCIe module X24 (X48) supports switching of 1-4 PCIe lanes for up to 6 (12) AMC slots.
Several configuration options which can be set by the MCH configuration menu (see Chapter *Configuration*) allow the flexible adoption of the PCIe hub to a certain environment.

The PCIe module V2.3 (Gen3) features 12 LEDs on the face plate driven by the FPGA to visualize the link status of the PCIe connections.

3.1.3.2 SRIO Switching Module (x24 and x48)

The SRIO module x24(x48) supports switching of 1 or 4 lanes SRIO for up to 6(12) AMC slots. The baud rate of each port can be selected independently between 1.25 Gbit/s, 2.5 Gbit/s and 3.125Gbit/s.

With the help of E-Keying the baud rate of the respective MCH SRIO port is configured according the supported baud rate of the connected AMC.

3.1.3.3 XAUI (10 Gigabit Ethernet) Switching Module (X24 and X48)

The XAUI module X24 (X48) supports switching of 10GbE for up to 6 (12) AMC slots and a 2nd MCH.

Additional the XAUI Module can be equipped with an optional face plate module (available in Q4 2008). With this module there are also two 10 gigabit Ethernet interfaces accessible at the face plate of the MCH.

There are two versions of the face plate module available a CX4 and a SFP+ version. The CX4 version offers two CX4 interfaces (10 GbE via copper cable). The SFP+ version offers two SFP+ cages. These cages can be assembled with any SFP+ transceiver. There are different SFP+ transceivers available, from different vendors for different optical interfaces.

If a compliant AMC is recognized via E-Keying the dedicated port of the switch is automatically enabled by the firmware; thus the switch is in a forwarding state.

4 Operation

The NAT-MCH should be operated in a MTCA R1.0 compliant uTCA shelf with forced air cooling only. This manual describes the operation of the following NAT-MCH PCB versions and firmware releases:

NAT-MCH V2.1	***Firmware V2.1 and later***
NAT-MCH V3.2	***Firmware V2.6 and later***
NAT-MCH V3.3	***Firmware V2.6 and later***
NAT-MCH V3.4	***Firmware V2.6 and later***
NAT-MCH Base-M4	***Firmware V2.14 and later***

4.1 System Startup and Redundancy Operation

After power-up the NAT-MCH tries to read the backplane FRU information and locates the available power modules. The NAT-MCH determines the primary and the secondary power module according to the role of the power module. In parallel the MCH tries to locate a second MCH in the system and if one exists, it starts the negotiation process to assign primary and backup roles to the MCHs.

If a second MCH is found in a redundant system, the primary MCH immediately starts backing up its internal database to the redundant MCH on a frequent basis via the GbE backplane link. The clock and the hub module on the redundant MCH are configured by the primary MCH and are running in active standby mode. The MCH software and hardware support fast fault detection and automatic management switchover.

To run the MCH in redundant configuration the Gigabit Ethernet option (-GbE) is mandatory.

4.2 AMC Module Startup Sequencing

After the NAT-MCH has completed its initial startup sequence it scans the system for available AMC modules. For any slot which is populated by an AMC module, a green LED on the face plate is lit. For all AMC modules found in the system, the MCH reads in the FRU information and sensor data records. If power negotiation is successful, it directs the power module to power up the AMC modules either in the activation sequence defined in the "Carrier Activation and Current descriptor" record of the backplane FRU device or - if the record is not found - according to the site number.

4.3 Local Shelf Manager

The local Shelf manager of the NAT-MCH provides management of the following resources within an uTCA system:

- Sensor Event Log (SEL)
- Temperature management and Cooling Unit control

4.3.1 Sensor Event Log

The MCH provides a System Event Log (SEL) that stores all events that occur in an uTCA system. The stored events are kept in the MCH's DRAM and are not stored into persistent memory.

Events can be read by an application using the IPMI message `GET_SEL_ENTRY_REQ`. In the MCH's default configuration events are removed from the SEL on read. Keeping events in the SEL after reading them can be configured using the MCH configuration menu (refer to chapter 5). To remove single events from the SEL in this configuration IPMI message `DELETE_SEL_ENTRY_REQ` has to be sent to the MCH, to clear the whole SEL IPMI message `CLEAR_SEL_REQ` has to be sent to the MCH.

Please note that some events (e.g. temperature events) are handled by the local shelf manager (refer to chapter 4.3.2).

4.3.2 Temperature Management

The local shelf manager receives temperature events from the following sources

- Local temperature sensors on the MCH
- Temperature sensors on the AMC modules
- Temperature sensors on Cooling Units and Power Modules

In case the MCH receives a temperature event (temperature going high event) from a FRU, i.e. the temperature of a certain module has reached a critical level, it increases the fan speed of the cooling units to the maximum level and starts monitoring the temperature sensors of the respective FRU. As soon as the temperature returns to normal level the fan speed will be decreased to a level which is 10% higher than the level it was when the temperature event occurs. The initial normal fan level can be set in the MCH configuration menu by the value of the configuration parameter:

<Default Fan Level>

Within the MCH global parameter section

4.4 NAT-MCHs operating with unmanaged Power Modules

The NAT-MCH is capable to manage systems which use so called "power through" modules. In this case the detection of modules is not done by the power modules (i.e. presence of #PS1 signal), but the NAT-MCH itself starts scanning all AMC slots by IPMI messages. Of course, due to the limited functionality of such a power module, functionalities like power sequencing or hot swap are not available.

5 MCH Configuration

The operation of the NAT-MCH can be adapted to certain environments by configuration options. The basic idea behind the configuration options is that a customer usually has to touch as less parameters as possible. Therefore, in most cases the default configuration will work.
The MCH configuration parameters are divided into several sections according to functionality:

- **Global parameter** – contains configuration values for basic MCH operation
- **Shelf Manager parameter** – configuration options for the local Shelf manager
- **Carrier Manager parameter** – configuration options for the Carrier manager
- **SEL parameter** – Sensor Event Log configuration
- **GbE switch parameter** – configuration options for the onboard Fabric A GbE switch
- **CLK module parameter** – configuration options for clock module (optional)
- **PCIe/SRIO parameter** – configuration options for the PCIe/SRIO Hub module (optional)
- **NTP parameter** - configuration options for optional **N**etwork **T**ime **P**rotocol support
- **DHCP parameter** – configuration of the built-in DHCP client

5.1 Configuration via console port

All configuration parameters are stored in the NAT-MCH configuration record in an onboard I2C-EEPROM. In order to change the configuration, the MCHs console port needs to be connected to a host computer.

The **NAT-MCH Gen2** provides the console port via a RS232 interface. A VT100 type terminal or a PC running a terminal program (e.g. TeraTerm, minicom etc.) has to be connected to the console port (19200, 8, N, 1) by the supplied cable.

Important Note: Despite the connector type may indicate that the console port of the NAT-MCH ***Gen2*** is an USB port, it is realized as a RS232 interface. Never connect this port to the USB port of a computer or to a hub. Unpredictable damage might be the result.

The **NAT-MCH Gen3 or –LC** provides the console port via a standard USB interface. The MCH identifies itself as a CDC type device and is supported by the standard "usbser.sys" driver within Windows2000/XP/Vista. An appropriate configuration file can be downloaded from N.A.T.s web site or FTP server (file "nat_mch.inf"). After the connection has been recognized by the USB driver, a terminal program (e.g. TeraTerm, minicom etc.) can be used via the assigned virtual COM port.

Pressing <return> displays the "nat>" prompt at the console.

The MCH is providing a low level command line interface (***CLI***) which allows to set certain operational parameters and to display run time information from the MCH and the system. Entering "?" will display a list of available commands.

For displaying and modifying the MCH configuration the following commands have been implemented:

mch	- prints all configuration settings
mchcfg	- menu based utility to set and modify configuration parameters
ip	- Basic Network configuration (IP addresses)

Parameters which should keep their values can be acknowledged by simply hitting ***<return>.*** For details about the MCH configuration please refer to chapter 5.4.

5.2 Configuration via the web Interface

All configuration parameters are accessible via the web interface as well. Please refer to Chapter 9 for the usage of the web interface. It can be necessary to setup a basic configuration like the IP address of the MCH before the web interface can be used.

5.3 Customizing the Network Configuration

The MCH requires IP address parameters to be adapted to make it working in a company's network environment.

Entering ***<ip>*** at the command line will show the actual configuration and allow to change these parameters by line editing the displayed values. IP address parameters must be entered in "xxx.xxx.xxx.xxx" form factor.
If an IP address is configured to 0.0.0.0 it will be ignored by the MCH. In case a non-zero gateway IP address is configured the related routing configuration will be performed automatically when the MCH starts up.

After all changes are done the system asks for confirmation of the new configuration. If the new values shall be written into the I2C EEPROM, the question should be answered with <y> and the new values will be become effective after the next power cycle.

Please refer to Appendix B also for more information about the IP address configuration.

5.4 Displaying and changing MCH operational parameters

Entering <***mch***> at the command line will show a list of operational parameters which allow to adapt the MCH to certain environments and configurations. With the command <***mchcfg***> it is possible to change these parameters.
The command <***mchcfg***> will lead to a menu driven utility which offers to modify the parameters within the individual sections according to their functionality.
Currently the configuration menu contains the following entries:

```
[ 0] no action
[ 1] print complete configuration
[ 2] reset to defaults
[ 3] modify MCH global configuration
[ 4] modify ShM configuration
[ 5] modify CM configuration
[ 6] modify SEL configuration
[ 7] modify GbE switch configuration
[ 8] modify CLK module configuration (optional)
[ 9] modify PCIe/SRIO configuration (optional)
[10] modify NTP configuration
[11] modify DHCP configuration
[ ?] print menu
[ h] print menu
[ q] quit and save configuration
```

Entering the respective number will guide to the associated configuration menu. ***[2] reset to defaults*** will reset all configuration parameters to a well-known "healthy" setup.

The following sections will describe the individual configuration options in detail.

5.4.1 MCH Global Parameters [3]

The MCH global parameter section contains basic operational and interface settings:

Configuration Option	*default*	*Description*
Management interface at GbE port	disabled	Allows re-routing of the management interface for RMCP and telnet access to the GbE Uplink port on the face plate. Default is using the 100BaseT port. (*, ***)
RMCP access	enabled	Allows to setup RMCP connections via the management port. Required for tools like openHPI, ipmiTool, NatView
Telnet access	enabled	Allows remote login to the MCH via telenet
SSH access	disabled	Allows remote login to the MCH via SSH
WEB access	enabled	Allows access to the MCH from any browser
IP address source Mgmt	board configuration	Source of the MCHs own IP address on the management port. Valid options are: • ***board configuration*** –saved value from local EEPROM – see chap. 5.3 • ***DHCP*** • ***ShM Link record*** – IP address is taken from Carrier FRU device, Shelf FRU info record • ***CM IP link record*** – IP address is taken from Carrier FRU device, Carrier FRU info record (**)
IP address source GbE	board configuration	Source of the MCHs own IP address on the Gigabit Ethernet port. Valid options are: • ***board configuration*** –saved value from local EEPROM – see chap. 5.3 • ***DHCP*** • ***ShM Link record*** – IP address is taken from Carrier FRU device, Shelf FRU info record • ***CM IP link record*** – IP address is taken from Carrier FRU device, Carrier FRU info record (**, ***)
RMCP session activity timeout minutes	0 min	Timeout for remote RMCP sessions being inactive – minutes
RMCP session activity timeout seconds	60 sec	Timeout for remote RMCP sessions being inactive – seconds
Default Fan Level	30%	Default fan level in percent after power up. A value of 0 remains the default value of the cooling unit active.
Enable Watchdog Timer	No	When enabled the MCH will be reset when an MCH firmware exception occurs, an important MCH firmware task is lost or the MCH stuck in an endless loop. After the reset the MCH will do a complete shutdown to ensure that everything works again. You can configure the Watchdog Timer in more detail via the MCH script. (s. chapter 10)
Enable backward compatibility	No	Enables backward compatibility to firmware version less or equal V2.4. If this mode is enabled CM and ShM behave as to use common SDR repository and SEL and single bridged IPMI messages are directly delivered to the target devices (PMs, CUs, AMCs,

		Clock/HUB modules). If this mode is disabled single bridged IPMI messages are delivered to the CM and target devices have to be accessed using double bridged IPMI messages which will be extracted first by the ShM and then by the CM.
Enable alternative cooling scheme	No	Refer to Appendix E
Control rear transition module fans	No	Enable control of RTM fans.
PM assignment strategy	strict	Strict: assign PMs always with the role they have in the backplane FRU after replacement of a faulty PM Relaxed: assign PMs to a not used role

Remark (*)
An external Shelf or System Manager can be connected to the NAT-MCH by an Ethernet connection running the RMCP protocol.
By default the management port is routed to the front panel 100 BaseT port.
Alternatively the management port can be switched to one link of the onboard GbE Switch. If the management port resides on the GbE switch, it can either be connected to an external shelf- or system manager by the GbE uplink port, or to any shelf or system controller running on one of the AMC modules, connected to the MCH by a GbE backplane link.

Remark (**)
Although it is possible to configure both the management and the Gigabit Ethernet ports to obtain its IP configuration from DHCP this approach is not useful: both Ethernet interfaces will be located in the same subnet and it is inexplicit to which physical interface packets are routed. Such a configuration will lead to discarding of packets.
Also note that DHCP is only allowed for the Ethernet port that is used for the management interface (refer to the option 'Management interface at GbE port'). This means if the option 'Management interface at GbE port' is set to disabled DHCP can only be used for the 'IP address source Mgmt'. If the option is enabled DHCP can only be used for the 'IP address source GbE'.

Remark (***)
Not available on NAT-MCH Gen3.

5.4.2 Shelf Manager Configuration [4]

The following configuration parameters control the operation of the MCH's onboard local shelf manager.

Configuration Option	*default*	*Description*
Allow Shelf FRU invalid	Yes	Allows startup and operation of the MCH even though no valid Shelf FRU record is found on the backplane I2C device.
Temperature management	Enabled	Allows the local Shelf manager to adapt the fan speed based on temperature events from the AMCs
Emergency shutdown	disabled	FRU on critical / non-recoverable event: Shelf manager shuts down an AMC if a threshold based sensor reaches the critical or non-recoverable limit. SYSTEM on critical / non-recoverable event: Shelf manager shuts down all AMCs if a threshold based sensor reaches the critical or non-recoverable limit.
Send SEND_MSG confirmation to SMS	no	If set to 'yes' the Shelf Manager immediately returns a SEND_MSG response to the BMC as a confirmation that the SEND_MSG command was received. This SEND_MSG confirmation does not contain response data to the embedded request. As soon as the Shelf Manager receives the response from the targeted destination the response is embedded into another SEND_MSG response that is delivered to the BMC. The BMC can distinguish between the first and the second received SEND_MSG by checking the data contents of the received SEND_MSG response. Please refer to IPMI Specification V2.0 sections about bridged messages for more details.
Use external Shelf Manager	no	Allows an external Shelf manager to operate via the RMCP interface. If enabled, the local shelf manager is suspended. Please note that usage of an external Shelf Manager is not yet supported.

5.4.3 Carrier Manager Configuration [5]

The carrier manager controls the communication and operation of the AMCs, power modules and cooling units.

Configuration Option	*default*	*Description*
Carrier number default	0	Allows the setting of a Carrier Number. If the value is 0 the Carrier number is loaded from the Carrier FRU device or from backplane I/O Expander. (*)
Quiesced event timeout	10	Timeout in seconds the MCH waits for a quiesce event from a FRU device in response of IPMI FRU CONTROL message "Quiesced" (**). The value of 255 defines an infinite timeout

Allow carrier FRU invalid	Yes	Allows the MCH to operate with internal default values if no valid backplane FRU device is found. If the flag is not set and the backplane FRU content is found invalid, the carrier manager will not enter normal operation mode.
Overrule carrier FRU	No	Overrules the backplane FRU device and uses internal defaults
Shutdown system if MCH goes down	No	Shutdown all resources of the system if the carrier manager of the MCH goes into M6 state.
Enable Clock E-keying	No	Enables Clock Source (if present) that matches a Clock Receiver of AMC (if present in Clock Configuration Record on AMC). Disables Clocks if AMC is removed. If the clock script is enabled, the Clock Source will not be enabled automatically. The MCH will check the script file and will enable the clocks if the script entries are valid.
Debugging Flags	0	Allows debugging or analyzing of certain areas of a uTCA system. (***)

Remark (*)
According to the uTCA specification the Carrier Number is defined either in the backplane FRU device or by DIP switches which can be read from an I/O expander at the I2C address 0x3e of the backplane I2C bus.
NAT has added a third option which can overrule the previous two by setting the carrier number in the MCH configuration record.

Remark (**)
If the handle of an AMC module is opened, the NAT-MCH transitions the module into state M6 and waits for the "Quiesced" event message. As modern CPUs and operating systems may take up to minutes to shut down completely, a configurable timeout has been added, which defines how long the MCH will wait for the "Quiesced" event message before it turns.

Remark (***)
If a management problem in an uTCA system occurs it might be extremely helpful to switch on certain debug options to identify the cause of the problem. Therefore the NAT-MCH allows setting debug levels for specific areas, like E-Keying, Cooling Unit management, etc.
Note: In a complex system it is easy to overload the system by debugs, especially if several debug options are turned on at the same time. As the display is flooded with message in this case it might become difficult to turn off the debugs again as access to the CLI is limited.
In such a case the system should be power cycled and the bootstrap process should be aborted by typing "z" immediately after the system has come up. This allows modifying the configuration parameters without disturbance from any debug output.

5.4.4 SEL Configuration Flags [6]

The SEL flags configure the behavior of the **S**ensor **E**vent **L**og (SEL).

Configuration Option	*default*	*Description*
Keep on Read	Disabled	If enabled, SEL entries are not deleted upon read
Allocate SEL in non-volatile RAM	yes	Allocate system event Log in non-volatile MRAM on Gen3 MCH
Ignore 'version change' sensor	yes	Ignore version change sensor if enabled.

5.4.5 GbE Switch Configuration [7]

The GbE Switch Configuration is used to configure the Ethernet switch located on the base board of the NAT-MCH. The different configuration options of the GbE switch are described in a separate manual (refer to Ethernet Switch Configuration Manual).

Configuration Option	*default*	*Description*
configuration source	none	Specifies the source for the GbE switch configuration. • no configuration • load from FLASH
Ignore Backplane FRU Info	no	Ignore backplane FRU information in switch management platform and assume a standard mapping scheme (use show_fruinfo 253 to display assumed backplane mapping).

5.4.6 CLK Module Configuration [8] (optional)

The following options are used to configure the CLK module optionally mounted on the NAT-MCH. The clock module can be configured via a text based configuration file. For more information please refer to chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**.

Configuration Option	*default*	*Description*
configuration source	none	Specifies the source of the CLK module configuration.* • no configuration • load from FLASH

5.4.7 PCIe Switch Configuration [9] (optional)

By the PCIe configuration parameters the operation of the (optional) PCIe Hub module is controlled.

Configuration Option	*default*	*Description*
operating mode (cluster 1) **	transparent	Operation mode of the PCIe Switches (PCIe cluster 1)*: • default (transparent) • NT Intelligent Adapter mode • NT Dual-Host mode
Upstream transparent slot number (cluster 1) **	0	AMC slot number where the host CPU resides in (i.e. root complex). (PCIe cluster 1)*
Upstream non-transparent slot number (cluster 1) **	0	AMC slot number where the non-transparent host CPU resides in for dual host mode. (PCIe cluster 1)*
operating mode (cluster 2) **	transparent	Operation mode of the PCIe Switches (PCIe cluster 2)*: • default (transparent) • NT Intelligent Adapter mode • NT Dual-Host mode
Upstream transparent slot number (cluster 2) **	0	AMC slot number where the host CPU resides in (i.e. root complex). (PCIe cluster 2)*
Upstream non-transparent slot number (cluster 2) **	0	AMC slot number where the non-transparent host CPU resides in for dual host mode. (PCIe cluster 2)*
Upstream slot power up delay	5 sec	Delay applied to the slot where the upstream CPU (RootComplex) resides in. The delay is applied before payload power is turned on. (*)
PCIe hot plug delay for AMCs	0 sec	Delay applied to the slots where downstream AMCs reside in. The delay is applied after payload power is turned on. (*)
100 MHz spread spectrum clock ***	Disabled	If enabled the FCLK-A clock will be of spread spectrum type with 100 MHz means. If disabled the FCLK-A is a 100MHz fixed clock.
hot plug support	Disabled	Enables PCIe Hot Plug Support. Refer to Appendix G PCIe Hot Plug Support (optional) for details on how the MCH firmware is handling the hot plug signals.
PCIe early Ekey	Disabled	Executes the E-Keying before Payload power is applied
'no ekey' for PCIe	Disabled	The PCIe ports are only enabled during E-Keying from

		MCH Firmware version V2.15 up when the E-Keying information match. To get a downwards compatible behavior this can be deactivated by setting the parameter 'no ekey' for PCIe to option enabled. In this case the PCIe ports are always enabled regardless of the E-Keying information.
PCIe clustering **	Disabled	Enables PCIe clustering for PCIe x48 HUB modules. If PCIe clustering is enabled the mode, transparent upstream and non-transparent upstream port parameters can be set for each switch separately. Communication between the two clusters is not possible in this case!

Remarks (*)
The power-up sequence and delays defined by the backplane FRU info device might not be sufficient in all situations for systems with PCIe boards, as usually all I/O boards need to be ready before the CPU board may start its PCI scan.
Therefore an extra delay in the MCH configuration record can be defined for the PCIe upstream host.
The settings for the second PCIe cluster are only valid if "PCIe clustering" is enabled via the corresponding flag.

Remarks (**)
PCIe Gen1 (HUB PCIe PCB V1.x) only; for PCIe Gen 3 refer to chapter 9.1 PCIe-Hub Module (optional) PCIe Virtual Switch Configuration (optional)

Remarks (***)
The configuration option '100 MHz spread spectrum clock' is only applicable when the PCIe Hub module isn't assembled as option LOSC.

5.4.8 SRIO Switch Configuration [9] (optional)

By the SRIO configuration parameters the operation of the (optional) SRIO Hub module is controlled.

Configuration Option	*default*	*Description*
operating mode	No uplink	Operation mode of the SRIO Switches: • No uplink, dual x4 Interconnect (default) • Dual Uplink Mode • Single Uplink + BP Fabric Update
SRIO module configuration source	no configuration	Load SRIO module configuration: • no configuration: 0 • load from FLASH: 1

5.4.9 NTP Configuration [10]

The MCH offers two client applications to obtain the date and time over the internet:

1. Time Protocol (TP)

The Time Protocol is a network protocol defined in RFC 868 to provide a site independent and machine readable date and time. A client uses TCP or UPD port 37 to request the time as

32 bit integer representing the number of seconds since 1. January 1900 GMT from the server.

Since the TP functionality was superseded by NTP/SNTP several time server do not longer support this protocol and request packets may be lost without reply.

The Time Protocol client application on the MCH is 'rdate'.

2. Network Time Protocol (NTP)

Network Time Protocol (NTP) is a networking protocol defined in RFC 5905 for clock synchronization between computer systems over packet switched, variable latency data networks. NTP provides UTC including scheduled leap second adjustments. No information about time zones or daylight saving time is transmitted.

The Simple Network Time Protocol (SNTP) is a less complex implementation of NTP, using the same protocol but without requiring the storage of time states and adjustments over extended periods of time.

The NTP/SNTP client application on the MCH is 'ntpdate'. Since the MCH does not store time states and adjustments the differentiation between NTP and SNTP is obsolete on the MCH.

The NTP client of the MCH can be enabled and configured by the following parameters:

Configuration Option	***default***	***Description***
NTP server IP	0.0.0.0	IP address of the NTP server in dot notation.
NTP 'check for time' delay minutes	0	Time interval to request a time update from the time server. Minutes and hours are added to define the poll time.
NTP 'check for time' delay hours	0	
NTP local time offset	0	Local time offset to GMT in hours. This is an integer number between -12 and +12 which is added to the retrieved time to define the current time on the MCH. NOTE: Negative numbers of hours are displayed as 256 - <hours> in the MCH configuration.
configuration flags:		
NTP client protocol	TP	Selects the client for TP or NTP/SNTP protocol.
NTP client	disabled	Enables/disables time request client on the MCH.

5.4.10 DHCP Configuration [11]

The built-in DHCP client can be configured by the following parameter:

Configuration Option	***default***	***Description***
Hostname	""	Designated Hostname for this MCH

Note: The DHCP client is activated by setting the configuration option <*IP address source*> to "*DCHP*" in the global MCH parameters.

6 Updating the MCH firmware

Firmware updates are supplied to customers as a result of bug fixes or enhancements implemented by N.A.T..

To execute a firmware update the binary image supplied by N.A.T. (for Firmware 2.7 and later running, a TAR-file containing firmware images for several components is supplied) must be available on a TFTP server within the customer's network. The MCH must be connected to this network via the management Ethernet port and a terminal must be connected to the console port.

6.1 Firmware Update from within the running MCH firmware

The firmware can be updated while the normal MCH firmware is running. Real time behavior of the MCH may be slowed down during Flash update. To perform the update, continue as described below "firmware update".

6.2 Firmware Update from Bootloader

In case that the MCH firmware image is corrupted e.g. due to aborted update process, the firmware can be updated via bootloader.

The MCH bootstrap process is executed in two phases: after power up an initial bootstrap loader is started; this loader in turn starts the final MCH application firmware image. To initiate a firmware update the normal bootstrapping process needs to be stopped before the initial bootloader enters the second phase by typing 'x' on the console terminal during the initial bootstrap phase. When the bootloader is stopped the command line mode of the bootloader is displayed.
Note on MCH Gen3: This option does no longer exist for the MCH Gen3. The firmware upgrade can be executed from within the running MCH only.

6.3 Firmware Update via CLI

The firmware update is started by typing <update_firmware> at the command line prompt. The path information of the new binary image is needed; <ip-address> has to be written in the form factor "xxx.xxx.xxx.xxx":

<ip-address:/path/to/the/new/firmware/bin-file>

If the given information is correct, the new firmware image will be loaded and stored into the onboard Flash device.

Warning: Do not power cycle the system during this process!

When programming was successful, power cycle the system to boot the new firmware.

Problem tracking: In most cases the TFTP download does not work, file protections on the TFTP server are not set correctly or firewalls prevent the MCH from downloading the image.

6.4 Firmware Update via web interface (FW 2.7 or later)

If the MCH works with firmware 2.7 or later it is possible to update several components of the MCH (like firmware, bootloader, microprocessor etc.) via web interface. For this purpose, a TAR-compressed collection of firmware-updates is supplied by N.A.T..

Update procedure:

- Download TAR-file to local space
- Access the web interface, select "Update MCH" from the web interface and open the saved file
- Clicking "Upload" displays a table with the current version of each updateable component of the MCH as well as the new version provided by N.A.T.
- Select which components should be updated: if the version in the TAR-file is newer than the current version, the update is selected automatically. If the version provided in the TAR-file is older than the current version and "Update this device?" is checked manually the component will be downgraded.
- After selecting the components to be updated, click "Update" and **WAIT UNTIL UPDATE HAS COMPLETED**.

The update may take several minutes depending on which and how many components are upgraded. A notification is given, when the update is completed and successful.

Note: The update function works with TAR-compressed files provided by N.A.T. only. Uploading a binary firmware file or a zip-compressed firmware file in the web interface will not work. If no updateable components are visible in the second step, please re-check the file type.

7 Management Interface

For interfacing to an external shelf or system controller the NAT-MCH is equipped with an Ethernet based management port. The management port can either reside on the front panel 100 BaseT port or on the GbE Uplink port. On the ***MCH Gen3*** one of the GbE Uplink ports is the designated management port; by default this is port "GbE 1". The assignment can be changed in the MCH configuration record. An external shelf or system controller can communicate with the MCH by the exchange of IPMI messages. The IPMI messages are encapsulated in RMCP packets for transport via Ethernet. By this standard interface the user has access to a broad range of open source and commercial tools to access and manage the MCH.
The N.A.T. graphical system management JAVA application "N.A.T. NatView" utilizes this interface as well.

7.1 Software structure

The following figure gives a basic overview about software structure on the MCH.

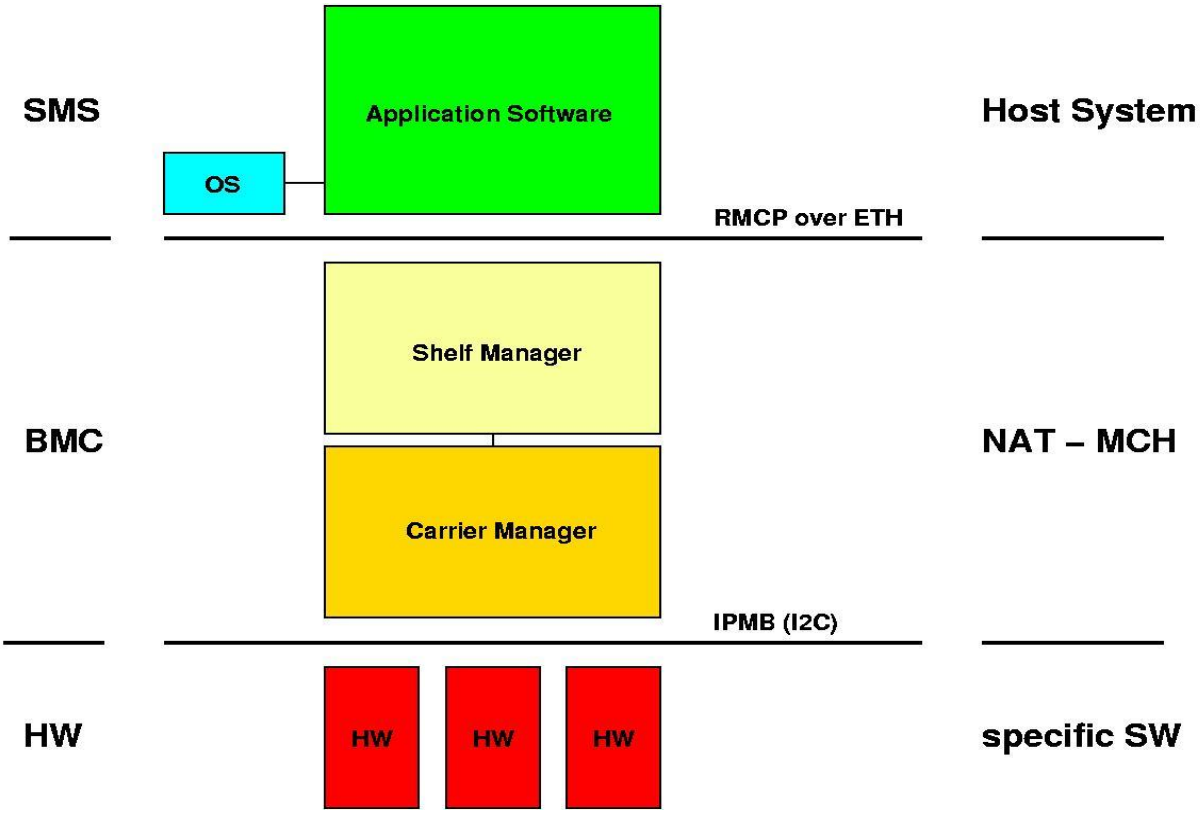


Figure 11 : Software structure overview

The System management software (SMS) is running on a host system that communicates via the Remote management control protocol (RMCP) via Ethernet with the MCH. The MCH itself implements base management controller (BMC) functionality and communicates via IPMI messages via I2C with hardware modules (e.g. Cooling Units, Power Modules, AMC cards) that run hardware module specific software.

The next figure gives an example for a software implementation that uses a graphical JAVA application running on top of the OpenHPI (see 7.3.3) middleware on the host system:

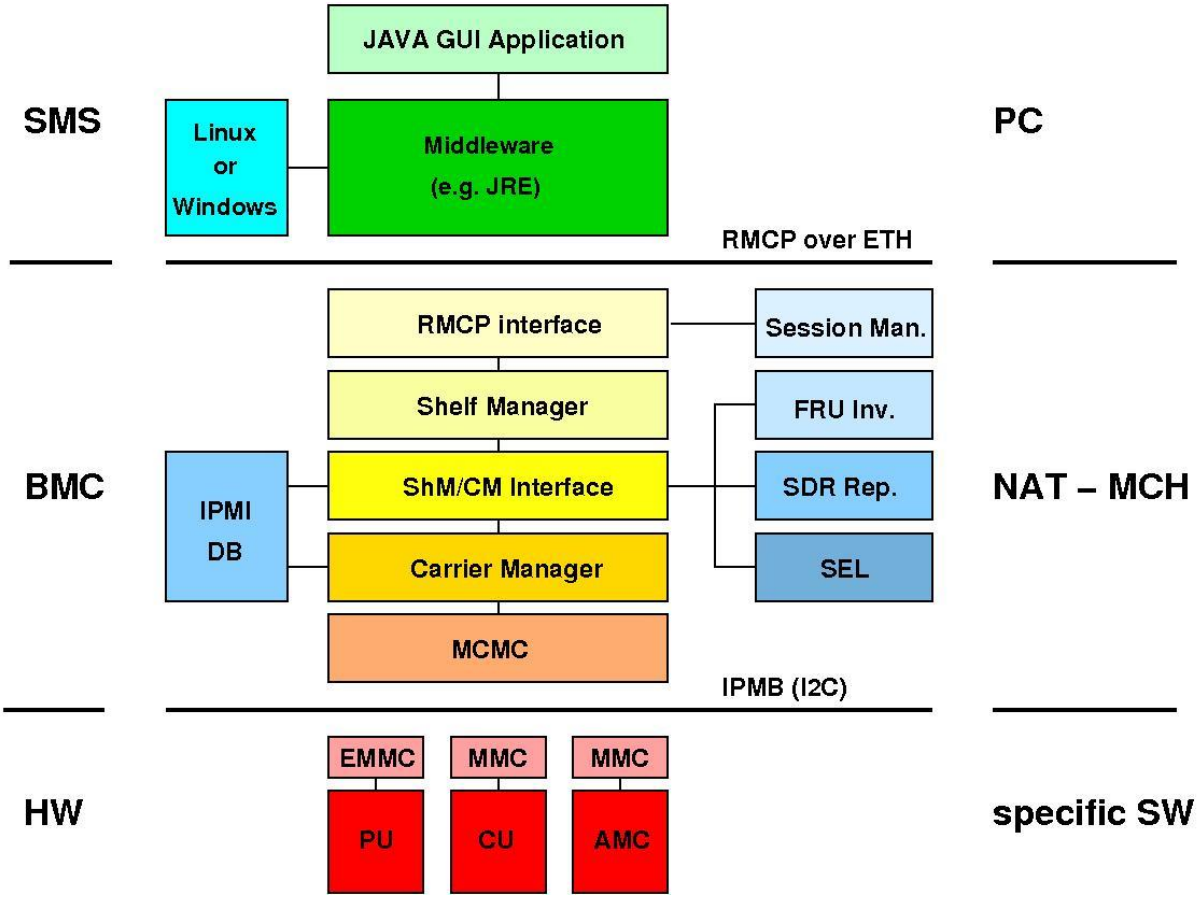


Figure 12 : Software structure details

The system management software is described more detailed in the following chapters.

The software on the MCH can be divided up into two parts: first a layer structured stack that implements the interfaces to external systems (host and hardware) and the message handling, and second helper modules to save data (IPMI data base, FRU inventory, SDR repository, SEL) and communication information (session management). Shelf Manager capability is optional and can be configured (see chapter 5.4.2) to allow usage of an external shelf

manager, e.g. on a host system or an AMC module. Note that the SEL on the MCH is kept in memory but not stored into persistent memory.

7.2 Communication between host system and MCH

Communication between a host system and the MCH is done via IPMI messages. To allow a flexible and common available transport of these IPMI messages the MCH offers an Ethernet based management interface.

7.2.1 Remote management control protocol

Transport over Ethernet usually requires some high level protocols to be used. The MCH implements a 'Remote Management Control Protocol' (RMCP, defined by the Distributed Management Task Force (DMTF, http://www.dmtf.org/) interface whereby the IPMI messages are embedded into the RMCP messages that are sent using UDP via an Ethernet channel.

The RMCP fulfills all requirements that are necessary to transfer IPMI messages over the Ethernet:

- Presence echo mechanism (RMPC ping/pong messages)
- Message flow control via sequence numbers
- Support for multi-session
- Transfer of session header
- Support for authenticated access
- Per-message authentication disable
- User-level authentication disable

The MCH supports access to the RMPC interface via its front Ethernet management interface. Since MCH firmware version 2.0 the MCH supports access to the RMCP interface via its backplane Ethernet interface. Selection between these two interfaces can be configured (refer to chapter 5.4).

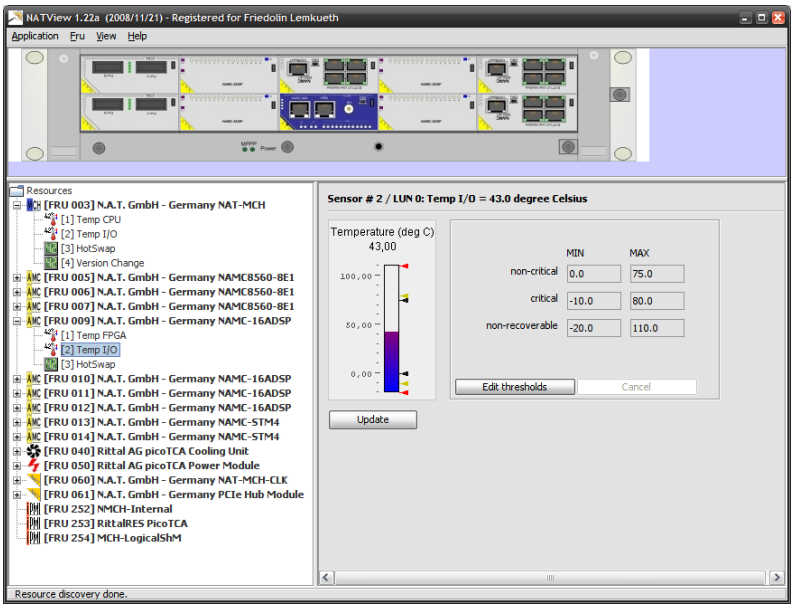
7.2.2 Supported IPMI messages

Handling of IPMI messages is done within a separate module of the software. The message module's configuration and a list of supported messages can be obtained using the CLI command 'imsg_info' (refer to chapter 8). This command allows printing the compile time configuration (list of supported message modules), the run time configuration and a list of request messages that are supported.

7.3 Host software overview

7.3.1 N.A.T. JAVA GUI application 'NATView'

NATView is a graphical application that can be used to monitor and control an uTCA system using IPMI messages. The application is written entirely in Java – it should therefore run on every operating system that is able to execute the Sun Java Runtime Environment (JRE).



7.3.1.1 Supported Java Releases

NATView requires Sun Java 1.6 or higher. This Sun Java version was chosen to guarantee the largest possible JRE installation basis on Window, Linux and Mac OS X systems. If in doubt enter

```
java -version
```

to a command prompt window. An appropriate copy of the Sun JRE can be obtained at **`java.sun.com`**.

7.3.1.2 Getting NATView

NATView can be downloaded from the N.A.T. FTP server **ftp.nateurope.com**. Username is **natmch**, password is **natmch** as well.

For NAT-MCH firmware 1.24 or higher, download latest released archive; for an older firmware version use NATView 1.25. Both archives contain all necessary data and documentation.

7.3.1.3 Installing and running NATView

NATView is usually distributed as a zip archive. The following steps will install NatView on the system:

1. Extract the content of the zip archives to an empty directory. This empty directory will be the root directory of the application.

2. Start the application **from within the root directory**. (Otherwise the application cannot find the board images in the subdirectory *images*.)
   From the command line type

```
cd <root-directory>
java –jar natview.jar
```

7.3.1.4 Further information

More detailed information can be found in the documentation that is accompanying the software.

7.3.2 Ipmitool

Ipmitool (available at http://ipmitool.sourceforge.net/) is an open source utility that runs on a Linux PC for managing and configuring devices that support the Intelligent Platform Management Interface. IPMI is an open standard for monitoring, logging, recovery, inventory, and control of hardware that is implemented independent of the main CPU, BIOS, and OS. The service processor (or Baseboard Management Controller, BMC) is the brain behind platform management and its primary purpose is to handle the autonomous sensor monitoring and event logging features.

The ipmitool program provides a simple command-line interface to this BMC. It features the ability to read the sensor data repository (SDR) and print sensor values, display the contents of the System Event Log (SEL), print Field Replaceable Unit (FRU) inventory information, read and set LAN configuration parameters, and perform remote chassis power control.

Ipmitool was not written to provide large-scale management application functionality. The functionality is easily accomplished by sending simple IPMI request messages and parsing the returned response. It is intended to be used by system administrators who like the simplicity and scriptability of command-line utilities, as well as those debugging or developing their own BMC implementations.

Please note that ipmitool is still under development. N.A.T. recommends to use at least version 1.8.11 since this version fixes some already known bugs (e.g. wrong decoding of SEND_MESSAGE responses).

7.3.2.1 ipmitool call syntax

The following list gives examples for ipmitool call syntax:

Common syntax to access the NAT-MCH over RMCP:

- `ipmitool -H <ip_address> -P "" <command>`

Common syntax to access a target device behind the NAT-MCH:

- `ipmitool -H <ip_address> -P "" -t <target_I2c> <command>`

Common syntax to access a target device behind the NAT-MCH using 'double bridged' OEM messages:

- `ipmitool -H <ip_address> -P "" -t <target_I2c> -T <target2_I2c> raw <nfn> <cmd> <data>`

where nfn is the network function code, cmd the IPMI message command and data the IPMI message data part given as hexadecimal values.

7.3.3 OpenHPI

This chapter gives a brief overview on how to install and start the OpenHPI for communication to the NAT-MCH. Please note that due to the availability of the JAVA base Natview tool N.A.T. support for making OpenHPI interworking with the NAT-MCH is obsolete now. However, N.A.T. continues performing tests using different OpenHPI versions.

OpenHPI is an open source project created with the intent of providing an implementation of the Service Availability Forum's Hardware Platform Interface (HPI, http://www.saforum.org/). HPI provides a universal interface for creating resource system models, typically for chassis and rack based servers, but extendable for other problem domains such as clustering, virtualization and simulation.

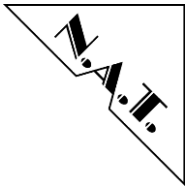
The current version of OpenHPI can be downloaded from http://www.openhpi.org/, documentation is available on http://openhpi.sourceforge.net/manual/book1.html. Please note that it is highly recommended to use the newest OpenHPI version that is available.

7.3.3.1 Overview OpenHPI

OpenHPI is used by N.A.T. as middleware to allow communication between the N.A.T. JAVA GUI HPI application and the NAT-MCH. Of course it is also possible to run other applications on top of OpenHPI.

OpenHPI runs on a standard Linux PC. N.A.T. tested the current OpenHPI version on Debian 3.1 (http://www.debian.org/), kernel 2.4.27 and on Debian 4.0, kernel 2.6.24 but it is known to run on other Linux distributions and other kernels as well. Once a Linux PC has been setup the preferred download directory should be changed and all distributed files should be copied into this directory:

- `cd <your_download_directory>`



- `cp <distributed_files> .`

Extract the provided OpenHPI tarball:

- `tar –xzvf openhpi-2.7.3.tar.gz`

and use the standard Linux mechanisms:

- `cd openhpi-2.7.3`
- `./configure && make && make install`

to install the software.

The following sections give some more detailed information on how to configure, compile and install OpenHPI. For more information about OpenHPI please refer to the README files that come with the OpenHPI package.

7.3.3.2 Configuring OpenHPI

By default, simply calling './configure' will build all plugins that **can** be built, all other plugins will be silently disabled. If a plugin should be disabled, any flags that are passed to configure will be passed to the 'configure' program.

For communication between OpenHPI and the NAT-MCH at least the plugin 'ipmidirect' must be enabled (this is the default in OpenHPI 2.7.3).

Please try

- `./configure –help`

for more information on the options. Further configuration information can be obtained from the README file provided in the openhpi-2.7.3 package.

7.3.3.3 Compiling OpenHPI

OpenHPI can be compiled by calling 'make'.

The official release of OpenHPI 2.7.3 contains a bug (that is fixed since 2.8.x versions):

./src/event.c:156: warning: `error' might be used uninitialized

this has to be fixed by replacing

```
int error;
```

by

```
int error = SA_OK;
```

in the file `./src/event.c'.

7.3.3.4 Installing OpenHPI

OpenHPI is installed by calling 'make install'.

Note that there is no daemon configuration file installed by default, so one has to copy it manually:

- cp ./openhpi.conf.example /usr/local/etc/openhpi/openhpi.conf

Alternatively the provided configuration file openhpi_nat.conf can be used for configuration:

- `cp ../examples/openhpi_nat.conf /usr/local/etc/openhpi/openhpi.conf`

7.3.3.5 Running OpenHPI

Before the OpenHPI daemon is started please start up the uTCA system. If the OpenHPI daemon is not able to establish a RMCP connection to the MCH it will silently terminate.

To start the OpenHPI daemon login as root on the Linux PC and call:

- `/usr/local/sbin/openhpid -c /usr/local/etc/openhpi/openhpi.conf`

Using the provided configuration file openhpi_nat.conf you will find daemon logging information on your system in /tmp/openhpi00.log.

As mentioned above the OpenHPI daemon will silently terminate on error conditions. If a RMCP connection cannot be established between the OpenHPI daemon and the MCH please note following:

- startup time can last up to 1 minute (depending on the number of found resources, i.e. sensors and FRU data
- check that the MCH can be reached over the network:
- `ping <MCH IP address>`
- if there is no ping reply please check the MCH's network configuration or the IP address and port settings in openhpi.conf
- check that the openhpi daemon is running:
- `ps aux | grep openhpi`

Once the daemon was started successfully one may start an application to communicate with the MCH. The OpenHPI package provides some applications, e.g.:

- `hpitree`

displays found resources,

- `hpitop`

displays resource information about found resources and

- `hpisensors`

displays information about found sensors.

NATView may also been used to communicate to the MCH and to display the found resources.

To stop the OpenHPI daemon please enter:

- `killall openhpid`

Please note that when stopping the OpenHPI daemon in that way open sessions will not be closed by the daemon. However, the MCH implements a 60 seconds inactivity timeout condition in its firmware that closes open sessions after this timeout.

7.3.3.6 Known issues with OpenHPI

N.A.T. tested most of OpenHPI versions since version 2.6.0 up to 2.14.0 but there are still open issues that have to be fixed:

- AMC hotswap does not work
- Event-Only sensors are not supported
- Resource detection is unreliable and changes when using different OpenHPI versions (e.g. physical devices are not detected)

7.3.4 Management Interface for NAT-JSM

This chapter gives a brief overview about user interface for NAT-JSM offered by NAT MCH.

7.3.4.1 Overrule Rotary Switch on NAT-JSM

The rotary switch position on the front panel of NAT-JSM can be overruled via MCH web interface. The JTAG-JSM menu offers drop down menu. Please, select new target device and confirm a choice with *Override* button.

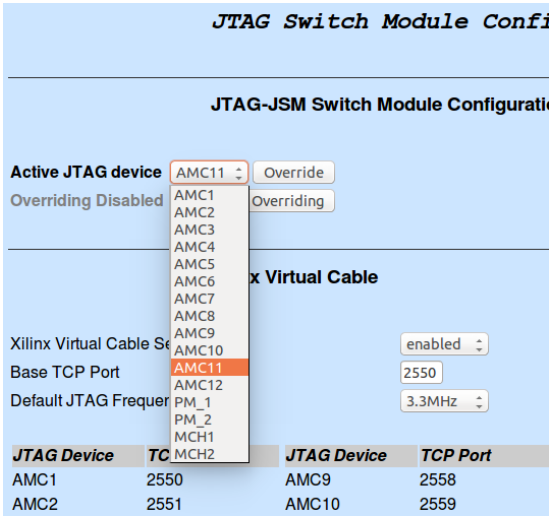


Figure 13: Overrule rotary switch position

The rotary switch overriding can be reset/disabled anytime. Please, press button "Reset overriding" to define current JTAG device via rotary switch state.

NOTE:
The changes are applied immediately, but are not saved permanently on board. After reboot or power cycle the current target device corresponds to rotary switch position.

7.3.5 Xilinx Virtual Cable Daemon

This chapter gives a brief Xilinx Virtual Cable daemon offered by NAT MCH.

The NAT-MCH provides TCP/IP based communicating interface for NAT-JSM. This is new server side application running on NAT-MCH to support Xilinx Virtual Cable (XVC) Protocol. The proprietary interface between the NAT-MCH and the NAT-JSM board allows for the Xilinx client software the JTAG access to the programming targets. Figure 14 illustrates the JTAG data flow from Xilinx software to the target JTAG device.

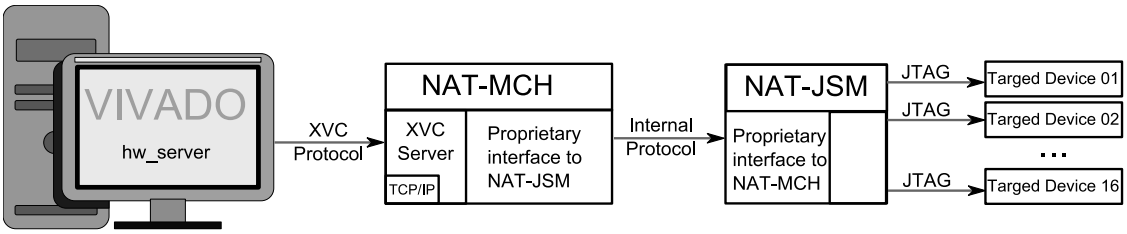


Figure 14: Block Diagram of XVC Connectivity Topology

NOTE: communication between NAT-MCH and NAT-JSM requires version 1.10 or later for FPGA of the NAT-MCH base board.

7.3.5.1 Base TCP port

The NAT-MCH keeps on listen 16 TCP ports for 16 programming targets. The first TCP port named *Base Port* is tunable and can be set/save by user. The other fifteen ports are automatically selected by incrementing the Base Port number. Therefore, each programming target becomes dedicated TCP port to be programmed by XVC client software. To view the mapping between TCP port and JTAG target, use, please, the menu *JSM* of the web interface on the NAT-MCH (Figure 15).

7.3.5.2 Default JTAG Frequency

The NAT-MCH generates the JTAG signals that are interconnected via NAT-JSM to a JTAG device. The signal frequency is controlled by XVC client. If the client doesn't control the frequency for whatever reason, the Default JTAG Frequency is to be applied. In this case the NAT-MCH offers the configuring interface of Default JTAG Frequency.

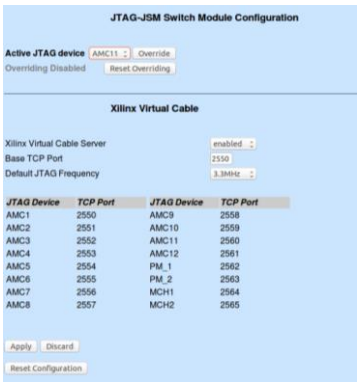


Figure 15: NAT-JSM menu

7.3.5.3 Reboot XVC-Server by Parameter Changing

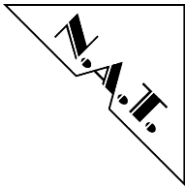
If the Base Port has and/or Default JTAG Frequency been changed, please, press the button *Apply* to confirm the change and save it permanently on board.

The changing of the Base Port and/or Default JTAG Frequency releases the restart of XVC Server. Therefore, all XVC connections will be aborted.

7.3.5.4 XVC-Client

By the time of writing of this chapter only three products of Xilinx were known for support of XVC: Vivado, Lab Tool and ChipScope Pro Analyse. The applications were tested for compatibility with NAT-MCH/NAT-JSM.

Further, an example shows the initiating of the connection between Vivado application and any AMC. To establish the connection, please, open Vivado Hardware Manager. Then start a Hardware Server session with following command in the Tcl Console:



>> connect_hw_server

After that, open hardware target with following *TCP* command in the Tcl Console:

>> open_hw_target -xvc_url <IP_Address>:< _Port>,

where <IP_Address> is IP address of a NAT-MCH and <TCP Port> is TCP port corresponds to particular target device.

In likewise manner the Lab Tool or ChipScope Pro Analyse can built JTAG connection over TCP via NAT-MCH

8 Command line interface

The NAT-MCH has a command line interface which allows users to initially set up configuration parameters as well as supplying low level diagnostic information. The command line interface is available at the console port.

The commands supported by the command line interface may vary with the firmware version. For an actual list of available commands enter "?" at the command line prompt. Please note that some commands are intentionally not listed in the table because they are used for testing purpose only.

The command line interface supports the following commands:

Command	Parameter	Description
lo		**loads a new firmware image** and stores in onboard Flash device (see chapter 6 for details). **NOTE:** This command is only available in the initial bootstrap loader.
bi		**Board Information** Prints the vital product information record (i.e. Serial number, Hardware revision and release codes).
br		**Baud Rate Configuration** Configures the MCH's serial interface baud rate by entering the desired baud rate, e.g. 19200 or 9600. The configured baud rate is stored in an on-board EEPROM and therefore valid after a reboot. Please note that if the baud rate was changed, the terminal's baud rate has to be re-configured too.
bs		**Boot String Configuration** This command allows to select between loading the MCH firmware from FLASH or to download a firmware image using TFTP. Default is to start the MCH firmware from FLASH.
ip		**IP configuration** Configures IP addresses, net mask, broadcast address and gateway.

telnetd_pw		**Telnet Password Configuration** (optional) Allows changing the password which is verified before starting a telnet/ssh session. A password length of not less than 8 to maximum 16 characters is required. Entering just <CR> without any other characters will delete the password. On the next telnet session start, no password verification will be executed. The ssh password will be reset to the default (user: root, pw: nat) in that case.
reboot		**Reboot the MCH**
update_firmware		**Update the MCH firmware in FLASH**
history		**Recall console history buffer (supported on MCH Gen3 only)**
update_fpga		**Update the FPGA firmware**
mch		**MCH configuration parameters** Shows the MCH configuration parameters as they are stored in the onboard I2C PROM. Parameters can be modified by [mmch].
mchcfg		**Modify MCH configuration parameters** **(see chapter** 5.4**)**
cmu_dbg		**Configure CM upper part debug**
csif_dbg		**Configure CM/ShM interface debug**
imsg_dbg		**Configure IPMI message debug**
lshm_dbg		**Configure local ShM debug**
red_dbg		**Configure redundancy module debug**
rmcp_dbg		**Configure RMCP interface debug**
sdrrep_dbg		**Configure SDR repository debug**
sel_dbg		**Configure System Event Log debug**
idb_info		**Print IPMI data base information**
imsg_info		**IPMI message information** Prints the implementation status of the supported IPMI messages on the Host (RMCP) interface and can be used to print a list of IPMI messages that are supported by the MCH.
lshm_info		**Print local ShM information**
sdrrep_info		**SDR repository information** Prints an overview of the Sensor Data Repository of the MCH.
sel_info		**System Event Log information** Prints the Sensor Event Log status and stored events in raw and decoded format.
session_info		**Session information** Print status of currently active sessions.
show_ekey		**Show all activated connections**
show_fru		**Show all FRUs**

show_fruinfo	fru_id	**FRU contents** Shows the contents of a FRU device selected by <fru_id>. For valid FRU numbers please refer to MTCA R1.0 table 3-3.
show_pwrconf		**Power Configuration** Displays the power channel assignment by the backplane FRU device
show_cu		**Show cooling unit**
show_pm		**Power Module Status** Shows the actual power allocation status for all AMC modules and Cooling Units.
show_sensorinfo	fru_id	**Shows the sensor values of the selected FRU**
version		**Print firmware version information**
ni		**Print network configuration**
arp		**Manipulate the system ARP** Use 'arp -?' to obtain a list of options.
dhcp_dbg		**Configure DHCP client debug**
dhcp_info		**Print DHCP client information**
ifconfig		**Print IP configuration**
ping	IP address	**Issue ICMP echo request** This command supports several command line options. For more information on the command's usage type 'ping -?'
rdate		**Get date from remote host** Gets the date from a time server (default: TCP port 37) and sets the date if desired. Use 'rdate -?'
sw_mp		**Ethernet switch management platform**
vlanp_cfg		**Port based VLAN configuration**
vlanq_cfg		**802.1Q VLAN protocol configuration**
vlanx_cfg		**802.1x VLAN security protocol configuration**
qos_cfg		**Quality of service menu** - priority configuration
qos1p_cfg		**802.1p Quality of service configuration**
mirr_cfg		**Ethernet port mirroring configuration**
show_gbe_links		**Print link states information about all XAUI ports (since V2.16)**
mac_amc		**Print MAC addresses for MAC slots ()** As MAC addresses are read from the MCH's gigabit controller only AMCs are detected which already have had IP traffic routed through the MCH's gigabit controller to that point in time. The mapping to the AMC slot number is done using the different ports of the gigabit controller.
show_swp2p		**Print all managed switch ports with the internal and physical mapping information**

show_xaui_links		**Print link states information about all XAUI ports (since V2.15)**
show_xmact		**Print XAUI MAC table (**since V2.16=>show_xaui_mact)
show_xaui_mact		**Print XAUI MAC table menu (since V2.16)**
show_xaui_stat		**Open status of XAUI switch menu.** Driver level – physical switch ports view.
diag		**Menu driven diagnostic tool** **This menu shall be used on NAT's explicit advice only!**
fan_ctl		**FAN control** Command to get fan properties and speed level and to set the fan speed level.
shutdown	<fru_id/all>	**Graceful shutdown of FRU or all AMCs**
shutdown	system system_hard	**Soft shutdown** of system (recommended) **Hard shutdown** (turn off power for all FRU immediately)
fru_start	<fru_id>	**Graceful start for FRU**

Table 8-1 List of CLI commands

9 Webserver

The NAT-MCH has an integrated embedded webserver which allows users to view and change configuration parameters of the NAT-MCH.

Before the webserver can be used it has to be enabled via the MCH configuration (refer to Chapter 0, "Web access").

The onboard webserver can be accessed with any standard web browser by entering the IP-address of the NAT-MCH's management port into the browser's address line. After the web browser has connected to the onboard webserver, the user is asked to enter a username and the associated password. The default values for these are:

Username: root
Password: nat

The functions supported by the webserver may vary with new firmware releases. Currently the following functions are supported.

Function	Sub Functions	Description
Setup Functions:		
Base Configuration	-	Provides a graphical configuration menu to change the basic configuration of the NAT-MCH corresponding to the command line interface command <*mchcfg*>.
Switch	-	The dropdown menu can be used to select the switch device to be configured.
Age Time	-	Provide a graphical configuration menu to change the Aging time for dynamically learned entries of the MAC table.
Port on/off	-	Provide a graphical configuration menu to change enable/disable state of an Ethernet switch ports. For more details please refer to the Ethernet Switch Configuration Manual.
Port VLAN	-	Provides a graphical configuration menu to change the port based VLAN settings. For more details please refer to the Ethernet Switch Configuration Manual.
802.1Q VLAN	-	Provides a graphical configuration menu to change the 802.1Q VLAN settings. For

Function	Sub Functions	Description
		more details please refer to the Ethernet Switch Configuration Manual.
802.1X	-	Provides a graphical configuration menu to change the 802.1X protocol settings. For more details please refer to the Ethernet Switch Configuration Manual.
802.1P	-	Provides a graphical configuration menu to change the 802.1P protocol settings. For more details please refer to the Ethernet Switch Configuration Manual.
Port Mirroring	-	Provides a graphical configuration menu to change the port mirroring settings. For more details please refer to the Ethernet Switch Configuration Manual.
Jumbo frame	-	Provides a graphical configuration menu to change the Jumbo frame settings. For more details please refer to the Ethernet Switch Configuration Manual.
Link Aggregation	-	Provides a graphical configuration menu to change the Link Aggregation settings. For more details please refer to the Ethernet Switch Configuration Manual.
Rapid Spanning Tree	-	Provides a graphical configuration menu to change the Rapid Spanning Tree settings. For more details please refer to the Ethernet Switch
SerDes/SGMII	-	Provides a graphical configuration menu to change the interfaces mode settings. For more details please refer to the Ethernet Switch
Link Status	-	Provides a graphical interface to view Ethernet link status.
BCM5395/FM2000/FM4000 Counters	-	Provides a graphical interface to view detailed port status.
Configure PCIe Virtual Switches (optional)		Provides a graphical configuration menu to change the PCIe Virtual Switch settings. For more details please refer to the chapter PCIe-Hub Module (optional) PCIe Virtual Switch Configuration

Function	Sub Functions	Description
		(optional).
PCIe Error Counter (optional)		Provides graphical counter overview for Receiver Errors, BadTLP errors and BadDLLP errors on a port of PCIe switch.
PCIe Link Status (optional)		Provides overview of current link speed and current link width of ports on PCIe switch.
SRIO GEN II - Status		Provides overview of current link speed and current link width of all ports on both SRIO switches.
SRIO GEN II - Routing Table		Provides Overview of Routing Tables
SRIO GEN II - Error Rate		Provides Overview of the Error Rate counters
SRIO GEN II - Port Statistic		Provides Overview of the Port Statistic counters
Maintenance Functions:		
Script Management		Backup current configuration settings to the onboard FLASH or an external file, or load settings from the onboard FLASH or an external file.
	Download Startup configuration	Can be used to download the current configuration file stored in the onboard FLASH memory.
	Download running configuration	Can be used to generate the current configuration (e.g. of the Ethernet switch or the clock module) as a text file. The configuration file will be generated from the current settings of the devices.
	Load startup-configuration	Load Startup configuration from FLASH memory and thereby overwrite the Running Configuration
	Delete startup-configuration	Delete Startup configuration from FLASH memory. It provides the loading of the default configuration on booting.
	Upload configuration file	Can be used to upload a configuration file. If the checkbox "*Save configuration to FLASH memory*" is checked, the new configuration will also be stored in the onboard FLASH memory.
	Verify	Can be used to verify a configuration file

Function	Sub Functions	Description
	configuration file	with the configuration stored in onboard FLASH memory.
Board Informations	-	Shows the hardware information and network configuration of the NAT-MCH. If there is a clock or hub mezzanine installed on the NAT-MCH, additional information about the hardware version(s) of these mezzanines is displayed as well.
Reboot NAT-MCH	-	Reboots the NAT-MCH. Please note that after reboot the connection has to be re-established manually via web-browser.
Update MCH		Opens a collection of firmware updates, shows current and new versions of each component and let user decide, which components to update. (refer to chapter 6.4)
N.A.T. Webpage	-	Opens a new browser window and loads the N.A.T. homepage.
Home	-	Displays the "Welcome" page of the NAT-MCH web based configuration interface.

Attention!
The most changes need a reboot of the NAT-MCH to apply. The webserver will show a notice if a reboot is necessary.

9.1 PCIe-Hub Module (optional)

9.1.1 PCIe Virtual Switch Configuration (optional)

The setup function Configure PCIe Virtual Switches provides a graphical configuration menu to change the PCIe Virtual Switch settings. It is possible to configure up to 6 Virtual Switches for clustering features (only 4 Virtual Switches on HUB-PCIe-x80; supported by MCH firmware from V2.16 onwards). For each Virtual Switch an upstream AMC port and the corresponding downstream ports can be selected separately. Furthermore a non-transparent AMC port can be configured on Virtual Switch 0.

Additionally a maximum link speed can be determined per PCIe switch port from MCH Firmware version V2.16 onwards. Usually a reduction isn't necessary. Nevertheless this feature can be helpful in systems in which transmission errors occur due to negotiating a too high link speed. In this situation the maximum PCIe link speed can be reduced on the port of an AMC that shall be limited.

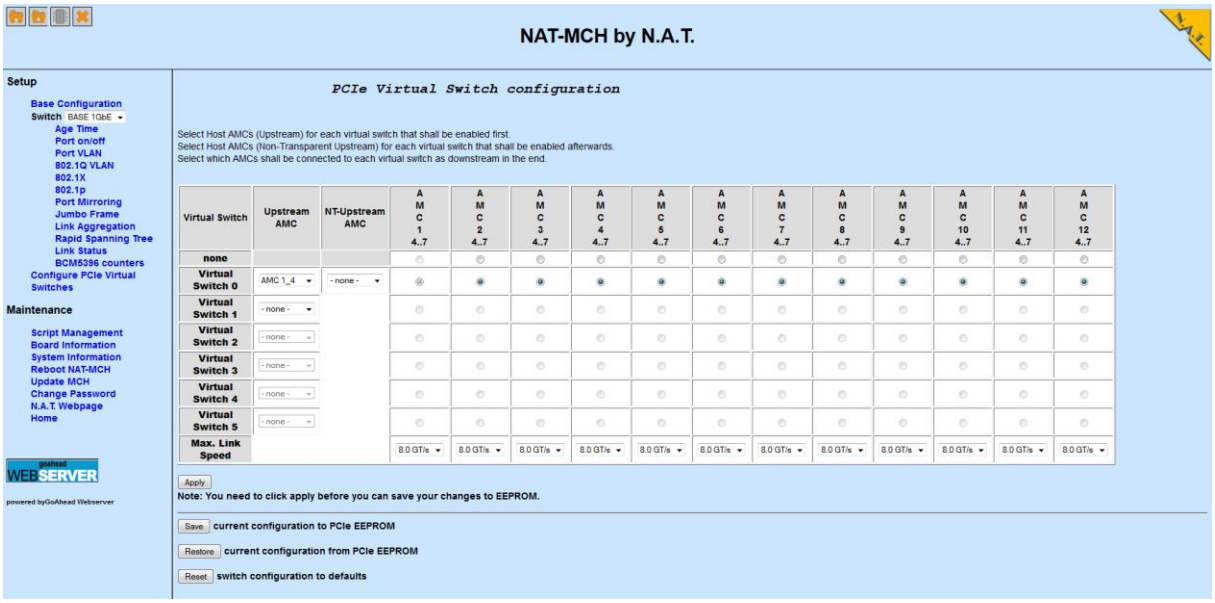


Figure 16 : NAT-MCH PCIe Virtual Switch Configuration via web-interface

Once the PCIe Virtual Switch Configuration is set up for its benefits the configuration can be stored permanently in the EEPROM. Therefore the button <Apply> has to be pressed first and afterwards the <Save> button. In case the configuration has been mixed up but the <Safe> button hasn't been pressed yet, the previous configuration can easily be restored by pressing the <Restore> button.

Note that the configuration might differ depending on the used backplane. Therefore it might be necessary to setup the configuration again when the NAT-MCH is used in a

system with a different backplane. The button <Reset> should be pressed first before setting up the configuration.

Further please note that operation of PCIe on AMC backplane Port 8-11 is not foreseen by the MTCA specification. Nevertheless this option is supported by the MCH firmware from V2.15 onwards and HUB-PCIe (PBC >= V2.1; AVR FW >= V1.6 and FPGA image >= V1.4).
From MCH firmware version V2.18 onwards the Link Width (e.g. x8 PCIe Link Width) can also be configured via the web-interface. The following conditions have to be fulfilled before this feature can be used:

1. backplane of the chassis needs to support this feature (e.g. like a NATIVE-C1 chassis, in which the port 4..7 and port 8..11 of an AMC are routed as one AMC port 4..11 on the fabrics of the MCH towards the right PCIe switch ports of the HUB-PCIe module).
2. x8 PCIe Link Width is only supported with a **HUB-PCIe-x48 (PCB >=V2.5)** or a **HUB-PCIe-x80** module (related PCIe switch ports have to be routed to an AMC when configuring a x8 link width)

In case these conditions are fulfilled the PCIe Link Width can be configured via the web-interface as shown in the following screen shot:

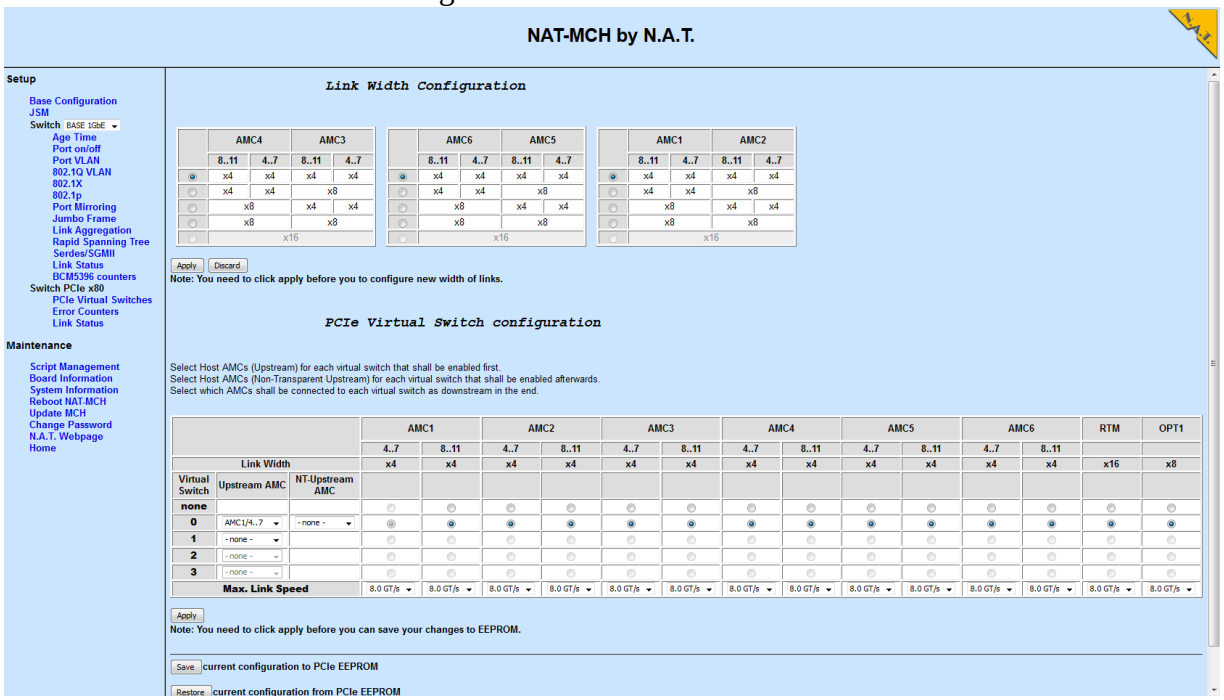


Figure 17 : NAT-MCH PCIe Width Link configuration via web-interface

The <Link Width Configuration> is only showing the configuration options that are supported in the system. For example the x16 configuration option is greyed out in the screen shot above since this feature isn't supported by the system. It needs to be considered that the AMC has to be configured separately (e.g. own BIOS settings or DIP switch settings) before the x8 link width is supported by that AMC.

9.1.2 PCIe Error Counter

PCIe Error Counter menu provide graphical counter overview for Receiver Errors, BadTLP errors and BadDLLP errors on a port of PCIe switch.

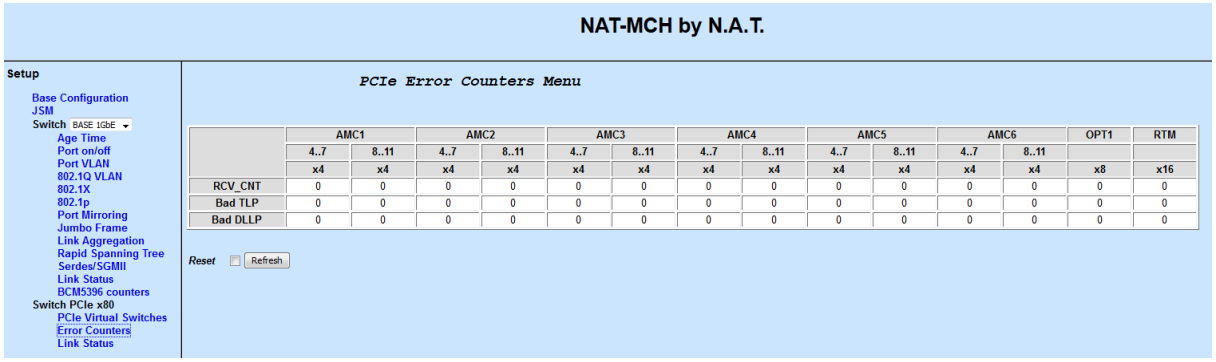


Figure 18 : NAT-MCH PCIe Error Counters via web-interface

The Counters are cleared with any "Refresh", if "Reset" checkbox has been set.

9.1.2.1 Receiver Error Counter

The counter contains four 8-bit fields that, when read, return the quantity of Receiver errors that the Port detected. The Error. Counter saturates at 255.

9.1.2.2 Bad TLP Counter

It counts the quantity of TLPs received with bad LCRC, or quantity of TLPs with a Sequence Number Mismatch error. The Counter saturates at 4294967295 (FFFF_FFFFh) and does not roll over to 0000_0000h.

9.1.2.3 Bad DLLP Count

The counter reports the quantity of DLLPs received with bad LCRC, or quantity of DLLPs with a Sequence Number Mismatch error. The Counter saturates at 4294967295 (FFFF_FFFFh) and does not roll over to 0000_0000h.

9.1.3 PCIe Link Status

The PCIe Link Status menu provide overview of current link speed and current link width of all PCIe switch ports.

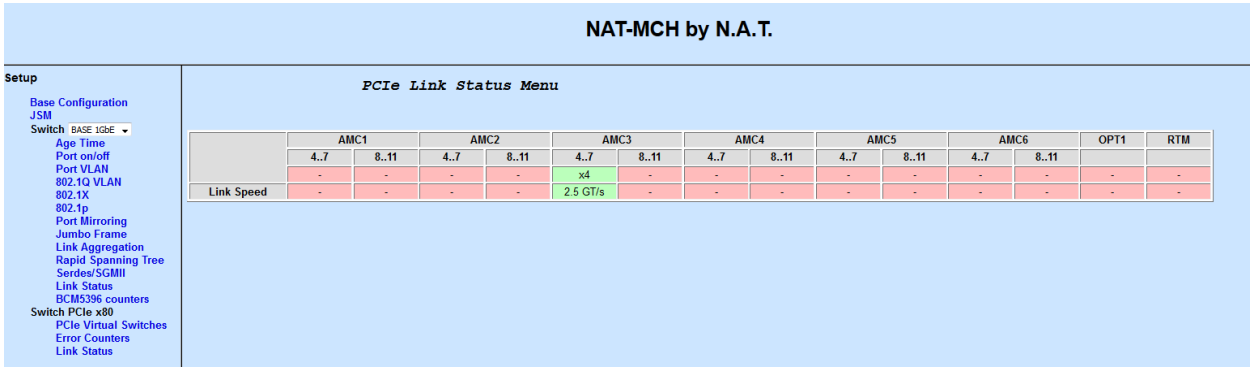


Figure 19 : NAT-MCH PCIe Link Status via web-interface

Negotiated Link Width: Dependent upon the physical Port configuration. Link width is determined by the negotiated value with the attached Lane/Port.

Current Link Speed: Indicates the negotiated Link speed of the Port's PCI Express Link.

9.2 SRIO-Gen II-Hub Module (optional)

9.2.1 Status Menu

The SRIO Link Status menu provide overview of current link speed, current link width and lane assignment on a link of all SRIO switch ports.

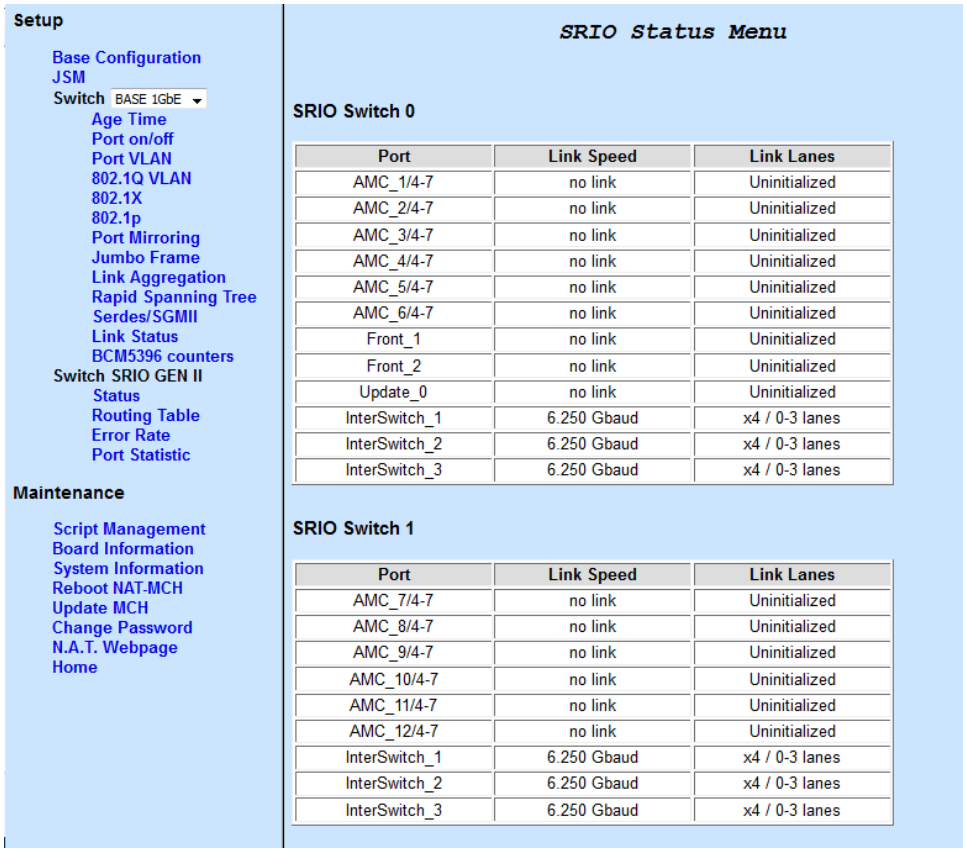


Figure 20 : NAT-MCH SRIO GEN II Link Status via web-interface

9.2.2 Routing Table

The main function of each SRIO port is to route received packets to the appropriate port(s) on the switch. Packet routing is supported in a RapidIO standard method using routing tables and standard RapidIO registers for multicast functionality.

Each SRIO port provides a 256 entry Device Routing Table and a 256 entry Domain Routing Table. The web interface menu provides configuration overview of routing tables of both SRIO switches on the SRIO-Hub module.

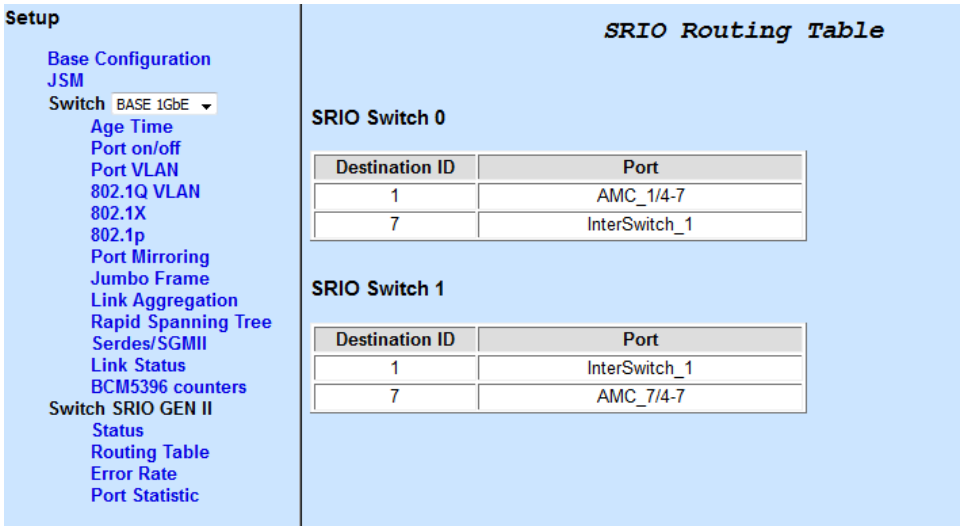


Figure 21 : NAT-MCH SRIO GEN II Routing Tables via web-interface

The menu shows all user defined entries. Other entries are set to default and have been hidden here.

9.2.3 Error Rate

Bit error rate menu can be used to count the number of received bit errors on the link.

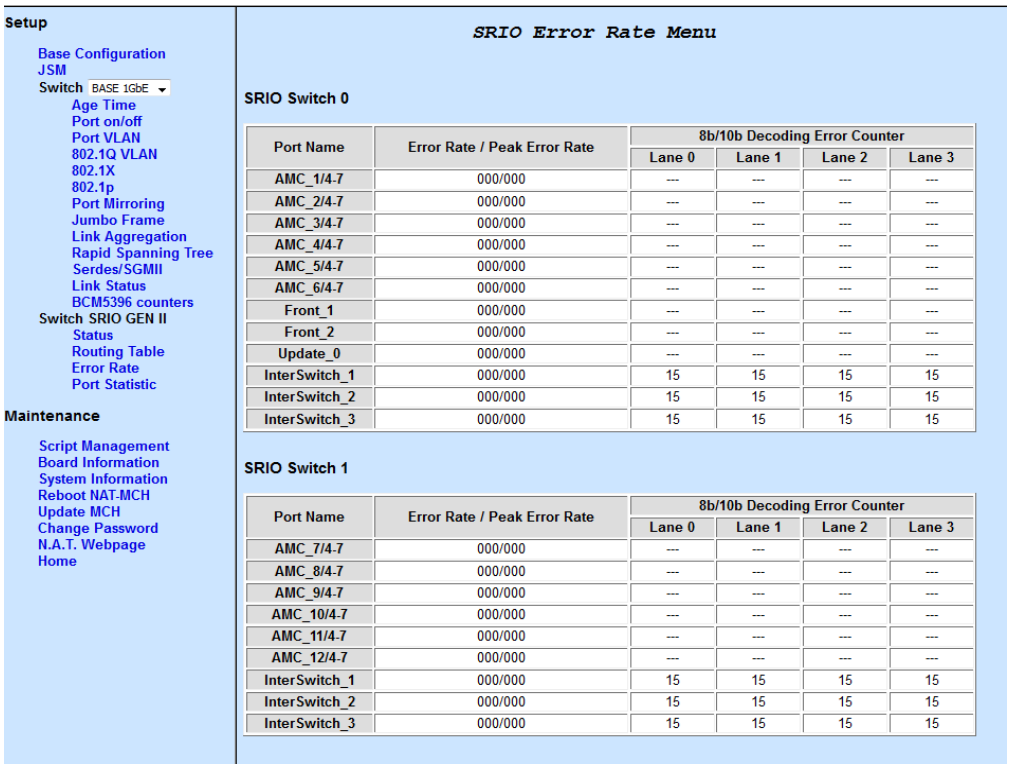


Figure 22 : NAT-MCH SRIO GEN II – Error Rate via web-interface

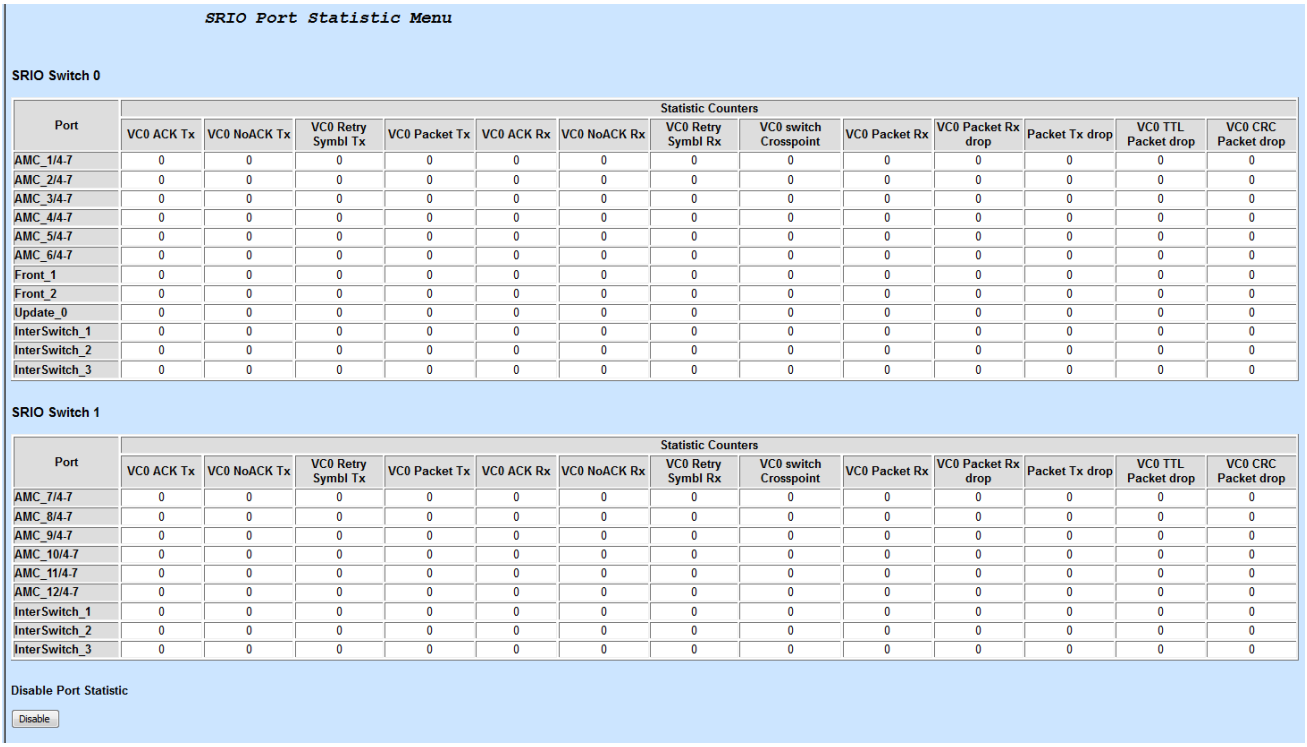
The *Error Rate* counts the number of transmission errors that have been detected by the port, decremented by the Error Rate Bias function to create an indication of the link error rate. This value does not clear on read.

The *peak value* attained by the error rate counter. This value does not clear on read.

8b/10b Decoding Error Counter indicate saturating count of 8b/10b decoding errors that have been detected for this lane since the field was last read

9.2.4 Port Statistic

The Port statistic menu provides a port statistic overview of all port on both SRIO switches of SRIO-Hub module.


SRIO Port Statistic Menu

SRIO Switch 0

Port	Statistic Counters												
	VC0 ACK Tx	VC0 NoACK Tx	VC0 Retry Symbl Tx	VC0 Packet Tx	VC0 ACK Rx	VC0 NoACK Rx	VC0 Retry Symbl Rx	VC0 switch Crosspoint	VC0 Packet Rx	VC0 Packet Rx drop	Packet Tx drop	VC0 TTL Packet drop	VC0 CRC Packet drop
AMC_1/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_2/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_3/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_4/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_5/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_6/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
Front_1	0	0	0	0	0	0	0	0	0	0	0	0	0
Front_2	0	0	0	0	0	0	0	0	0	0	0	0	0
Update_0	0	0	0	0	0	0	0	0	0	0	0	0	0
InterSwitch_1	0	0	0	0	0	0	0	0	0	0	0	0	0
InterSwitch_2	0	0	0	0	0	0	0	0	0	0	0	0	0
InterSwitch_3	0	0	0	0	0	0	0	0	0	0	0	0	0

SRIO Switch 1

Port	Statistic Counters												
	VC0 ACK Tx	VC0 NoACK Tx	VC0 Retry Symbl Tx	VC0 Packet Tx	VC0 ACK Rx	VC0 NoACK Rx	VC0 Retry Symbl Rx	VC0 switch Crosspoint	VC0 Packet Rx	VC0 Packet Rx drop	Packet Tx drop	VC0 TTL Packet drop	VC0 CRC Packet drop
AMC_7/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_8/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_9/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_10/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_11/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
AMC_12/4-7	0	0	0	0	0	0	0	0	0	0	0	0	0
InterSwitch_1	0	0	0	0	0	0	0	0	0	0	0	0	0
InterSwitch_2	0	0	0	0	0	0	0	0	0	0	0	0	0
InterSwitch_3	0	0	0	0	0	0	0	0	0	0	0	0	0

Disable Port Statistic

Disable

Figure 23 : NAT-MCH SRIO GEN II – Port Statistic via web-interface

The supporting counter are showing in the following table:

Counter	Description
VC0 ACK Tx	VC0 Acknowledgements Transmitted Counter
VC0 NoACK Tx	Not Acknowledgements Transmitted Counter
VC0 Retry Symbl Tx	VC0 Retry Symbols Transmitted Counter Register
VC0 Packet Tx	VC0 Packets Transmitted Counter
VC0 ACK Rx	VC0 Acknowledgements Received Counter

VC0 NoACK	Not Acknowledgements Received Counter
RxVC0 Retry Symbl Rx	VC0 Retry Symbols Received Counter
VC0 switch Crosspoint	VC0 Switch Crosspoint Buffer Output Packet Counter
VC0 Packet Rx	VC0 Packets Received Counter
VC0 Packet Rx drop	VC0 Received Packets Dropped Counter
Packet Tx drop	VC0 Transmitted Packets Dropped Counter
VC0 TTL Packet drop	VC0 TTL Packets Dropped Counter
VC0 CRC Packet drop	VC0 CRC Limit Packets Dropped Counter

10 Script Based Configuration

Several settings of the NAT-MCH can be changed via a text based configuration file. A script file containing the current configuration of the NAT-MCH can be downloaded via the web server interface (refer to chapter 9). This file can be edited using a standard text editor and uploaded again to the NAT-MCH.

The script contains so called "configuration items" which are used to setup specific parameters on the NAT-MCH. Each configuration item consists of an identifier and one or more parameters. The functionality and usage of each configuration item is described in the script file above the item itself.

Comments in the script file can be added using the hash character '#' e.g. like this:

```
# This is a comment
```

The following chapters describe the configuration parameters that can be set using the text based script.

10.1 Ethernet Module Configuration

The configuration of the 1 Gbe Ethernet switch located on the NAT-MCH board and the 10 GbE Ethernet located in the HUB module (if assembled) can be changed using the text based configuration script.

The configuration items available for the Ethernet switches are documented in the "Ethernet Switch Configuration Manual".

10.2 Clock Module Configuration

The different functions of the clock module can be set by using so called "configuration items". Each configuration item consists of an identifier and one or more parameters. The following example shows the configuration item "`clk_type`" which can be used to assign a specific clock type (e.g. 2.048 MHz) to one of the output clocks.

```
clk_type = 1, 3
```

As shown above the configuration item identifier is always separated from the parameter list by an equals sign ('='). The parameters are always separated by a comma sign.

Comments within the configuration begin with a hash character ('#'), like:

```
#
# Item <<clk_type>>: clock type configuration
#
```

The current clock module configuration can be downloaded from the NAT-MCH e.g. by using the "Script Management" function in the web based configuration interface (refer to chapter 9). The generated text file can be changed with a standard text editor and can be used as a starting point for a user defined configuration. The changed configuration can be uploaded again, e.g. via the web based configuration interface.

The clock module configuration can also be stored in the onboard FLASH memory, e.g. by using the "Save current configuration to onboard FLASH memory" function in the web based configuration interface. This can be used to load the configuration for the clock module during startup of the NAT-MCH. Note that loading the clock module configuration from FLASH has to be enabled via the MCH configuration (refer to chapter 5.4.6).

The following chapters describe the configuration items which can be used to configure the clock module. Note that the configuration items might differ depending on the clock module version. Therefore the description of the configuration items is divided in different subchapters below. For more information about the functionality provided by the clock module please refer to the latest Clock-Module Technical Reference Manual.

10.2.1 Clock Module Version 2.x

10.2.1.1 PLL Reference Input Configuration

Description:
This configuration item is used to configure the clock source for the reference inputs of the PLL on the clock module.

Syntax:

```
clk_pll_ref = ref, src, slot
```

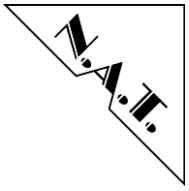
Parameter Description:

```
ref:      selects the reference input of the PLL
          1 - PLL input REF0
          2 - PLL input REF1

src:      reference source
          1 - CLK1
          2 - CLK2
          3 - CLK3
          4 - CLK1 UPDATE
          5 - CLK3 UPDATE

slot:     AMC slot number (1..12)
          only used if reference source is CLK1 or CLK2,
          else ignored
```

Example:

```
clk_pll_ref = 1, 2, 9
```

Use CLK2 from AMC slot 9 as input for PLL reference REF0.



10.2.1.2 PLL Mode Configuration

Description :
This configuration item is used to configure the operation mode of the PLL on the clock module.

Syntax :

```
clk_pll_mode = mode, ref
```

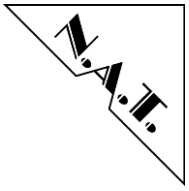
Parameter Description :

```
mode :    PLL operation mode
          1 - normal mode (use PLL reference input <ref>)
          2 - holdover mode
          3 - free running mode
          4 - automatic selection between REF0 and REF1

ref:      PLL reference input
          1 - PLL input REF0
          2 - PLL input REF1
          3 - PLL input REF2
```

Example:

```
clk_pll_mode  = 4, 1
```

Configure the PLL to automatically select the reference input between REF0 and REF1.



10.2.1.3 Clock Type Configuration

Description:
Determines the signal type for a specific clock type.

Syntax :

```
clk_type = clk, type
```

Parameter Description :

```
clk:        clock identifier
            1 - CLK1
            2 - CLK2
            3 - CLK3
            4 - CLK1 UPDATE
            5 - CLK3 UPDATE

type: clock signal type
            0 - NONE
            1 - 8 kHz
            2 - 1.544 MHz
            3 - 2.048 MHz
            4 - 3.088 MHz
            5 - 4.096 MHz
            6 - 6.312 MHz
            7 - 8.192 MHz
            8 - 8.448 MHz
            9 - 16.384 MHz
            10 - 19.44 MHz
            11 - 20 MHz (Stratum 3)
            12 - 32.768 MHz
            13 - 34.368 MHz
            14 - 44.736 MHz
            15 - 65.536 MHz
            16 - PCIe clock
            17 - PLL REF0 input
            18 - PLL REF1 input
            19 - EXT REF input
            20 - 100 us tick
            21 - SW_CLK
            22 - SYNC_CLK
            23 - HOLDOVER
```

NOTE: Refer to the Clock-Module Technical Reference Manual for a description of the different clock types.

Example:

```
clk_type  = 1,  3
```

Output a 2.048 MHz clock signal on CLK1.

10.2.1.4 Clock Output Configuration

Description:
Enables or disables the output of a specific clock type.

Syntax:

```
clk_out = clk, slot, ena
```

Parameter Description:

```
clk:      clock identifier
          1 - CLK1
          2 - CLK2
          3 - CLK3
          4 - CLK1 UPDATE
          5 - CLK3 UPDATE

slot:     AMC slot number (1..12) optional
          only used if reference source is CLK1 or CLK2,
          else ignored

ena:      0 = disable clock output
          1 = enable clock output
```

Example:

```
clk_out  = 1,  1, 1
```

Enable output of CLK1 on AMC slot 1.

10.2.2 Clock Module Version 3.x

10.2.2.1 PLL Reference Input Configuration

Description:
This configuration item is used to configure the reference inputs of the PLL on the clock module. The clock source that is provided to one of these PLL reference inputs has to be configured via the configuration item `clk_gen3_out` described below.

Syntax:

```
clk_gen3_pll_ref = ref, freq
```

Parameter Description:

```
ref:      selects the reference input of the PLL
          1 - PLL input IC1
          2 - PLL input IC2
          3 - PLL input IC3
          4 - PLL input IC4

freq:     input clock frequency in Hz
```

The following input frequencies are support by the PLL on the clock module:
2 kHz, 4 kHz, 8 kHz, 1.544 MHz, 2.048 MHz, 5 MHz, 6.312 MHz, 6.480 MHz, 10 MHz, 19.44 MHz, 25 MHz, 25.92 MHz, 31.25 MHz, 38.88 MHz, 51.84 MHz, 62.5 MHz, 77.76 MHz, 125 MHz, 155.52 MHz, 156.25 MHz.

Furthermore frequencies which are a multiple of 2 kHz are supported up to 125MHz or frequencies which are a multiple of 8 kHz. As the input clock has to be divided down to 2kHz or 8kHz in this case and there is only one divider available in the PLL chip, only one clock which is different from the standard clocks above can be configured (except the divider is the same for 2 or more input clocks).

Example:

```
clk_gen3_pll_ref  =  1,   10000000
```

A 10MHz clock is provided to PLL reference input IC1.

10.2.2.2 PLL Mode Configuration

Description :
This configuration item is used to configure the operation mode of the PLL on the clock module.

Syntax :

```
clk_gen3_pll_mode = mode, ref
```

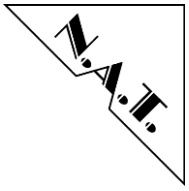
Parameter Description :

```
mode :    PLL operation mode
          1 - automatic selection between IC1 - IC4
          2 - forced mode (reference according to
              parameter ref)
          3 - free running mode

ref:      PLL reference input (only for forced mode)
          1 - PLL input IC1
          2 - PLL input IC2
          3 - PLL input IC3
          4 - PLL input IC4
```

Example:

```
clk_gen3_pll_mode  = 1, 0
```

Configure the PLL to automatically select the reference input between IC1 – IC4.



10.2.2.3 Clock Output Configuration

Description:
Determines which clock source is driven on a specific clock output of the clock module.

Syntax :

```
clk_gen3_out = dst, src
```

Parameter Description :

```
dst: destination clock identifier
      1 - CLK1 AMC  1
      2 - CLK1 AMC  2
      3 - CLK1 AMC  3
      4 - CLK1 AMC  4
      5 - CLK1 AMC  5
      6 - CLK1 AMC  6
      7 - CLK1 AMC  7
      8 - CLK1 AMC  8
      9 - CLK1 AMC  9
     10 - CLK1 AMC 10
     11 - CLK1 AMC 11
     12 - CLK1 AMC 12
     13 - CLK2 AMC  1
     14 - CLK2 AMC  2
     15 - CLK2 AMC  3
     16 - CLK2 AMC  4
     17 - CLK2 AMC  5
     18 - CLK2 AMC  6
     19 - CLK2 AMC  7
     20 - CLK2 AMC  8
     21 - CLK2 AMC  9
     22 - CLK2 AMC 10
     23 - CLK2 AMC 11
     24 - CLK2 AMC 12
     25 - CLK3 AMC  1
     26 - CLK3 AMC  2
     27 - CLK3 AMC  3
     28 - CLK3 AMC  4
     29 - CLK3 AMC  5
     30 - CLK3 AMC  6
     31 - CLK3 AMC  7
     32 - CLK3 AMC  8
     33 - CLK3 AMC  9
     34 - CLK3 AMC 10
     35 - CLK3 AMC 11
     36 - CLK3 AMC 12
     37 - CLK1 Update
     38 - CLK3 Update
     39 - EXT single ended 1
```

```
        40 - EXT single ended 2
        41 - EXT single ended 3
        42 - EXT single ended 4
        43 - EXT differential 1
        44 - EXT differential 2
        45 - PLL IC1
        46 - PLL IC2
        47 - PLL IC3
        48 - PLL IC4
        49 - PLL SYNC1
        50 - PLL SYNC2
        51 - PLL SYNC3
        52 - SYNC CLK input

src:  source clock identifier
         0 - disabled
         1 - CLK1 AMC  1
         2 - CLK1 AMC  2
         3 - CLK1 AMC  3
         4 - CLK1 AMC  4
         5 - CLK1 AMC  5
         6 - CLK1 AMC  6
         7 - CLK1 AMC  7
         8 - CLK1 AMC  8
         9 - CLK1 AMC  9
        10 - CLK1 AMC 10
        11 - CLK1 AMC 11
        12 - CLK1 AMC 12
        13 - CLK2 AMC  1
        14 - CLK2 AMC  2
        15 - CLK2 AMC  3
        16 - CLK2 AMC  4
        17 - CLK2 AMC  5
        18 - CLK2 AMC  6
        19 - CLK2 AMC  7
        20 - CLK2 AMC  8
        21 - CLK2 AMC  9
        22 - CLK2 AMC 10
        23 - CLK2 AMC 11
        24 - CLK2 AMC 12
        25 - CLK3 AMC  1
        26 - CLK3 AMC  2
        27 - CLK3 AMC  3
        28 - CLK3 AMC  4
        29 - CLK3 AMC  5
        30 - CLK3 AMC  6
        31 - CLK3 AMC  7
        32 - CLK3 AMC  8
        33 - CLK3 AMC  9
        34 - CLK3 AMC 10
```

```
35 - CLK3 AMC 11
36 - CLK3 AMC 12
37 - CLK1 Update
38 - CLK3 Update
39 - EXT single ended 1
40 - EXT single ended 2
41 - EXT single ended 3
42 - EXT single ended 4
43 - EXT differential 1
44 - EXT differential 2
45 - PLL OC1 (default 30.72MHz)
46 - PLL OC2 (default 10MHz)
47 - PLL OC3 (default 2.048MHz)
48 - PLL OC4 (default 19.44MHz)
49 - PLL OC5 (default 77.76MHz)
50 - PLL OC6 (default 155.52MHz)
51 - PLL OC7 (default 156.25MHz)
52 - PLL FSYNC 8kHz
53 - PLL MFSYNC 2kHz
54 - 12.8MHz OSC
55 - 25MHz OSC (only with HCSL option)
56 - LEVEL 0
57 - LEVEL 1
58 - SYNC_CLK
```

NOTE: Refer to the Clock-Module Technical Reference Manual for a description of the different clock types.

Example:

```
clk_gen3_out = 1, 50
```

Output the PLL clock output OC6 on CLK 1 of AMC slot 1.

10.2.3 Clock Module Version 4.x

10.2.3.1 PLL Reference Input Configuration

Description:
This configuration item is used to configure the reference inputs of the PLL on the clock module. The clock source that is provided to one of these PLL reference inputs has to be configured via the configuration item `clk_gen4_out` described below.

Syntax:

```
clk_gen4_pll_ref = ref, freq
```

Parameter Description:

```
ref:      selects the reference input of the PLL
          1 - PLL input IC1
          2 - PLL input IC2
          3 - PLL input IC3
          4 - PLL input IC4
          5 - PLL input IC5
          6 - PLL input IC6
          7 - PLL input IC7
          8 - PLL input IC8

freq:     input clock frequency in Hz
```

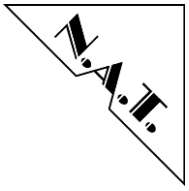
The following input frequencies are support by the PLL on the clock module:
2 kHz, 4 kHz, 8 kHz, 1.544 MHz, 2.048 MHz, 5 MHz, 6.312 MHz, 6.480 MHz, 10 MHz, 19.44 MHz, 25 MHz, 25.92 MHz, 31.25 MHz, 38.88 MHz, 51.84 MHz, 62.5 MHz, 77.76 MHz, 125 MHz, 155.52 MHz, 156.25 MHz.

Furthermore frequencies which are a multiple of 2 kHz are supported up to 125MHz or frequencies which are a multiple of 8 kHz. As the input clock has to be divided down to 2kHz or 8kHz in this case and there is only one divider available in the PLL chip, only one clock which is different from the standard clocks above can be configured (except the divider is the same for 2 or more input clocks).

Example:

```
clk_gen4_pll_ref = 1,   10000000
```

A 10MHz clock is provided to PLL reference input IC1.



10.2.3.2 PLL Mode Configuration

Description :
This configuration item is used to configure the operation mode of the PLL on the clock module.

Syntax :

```
clk_gen4_pll_mode = pll#, mode, ref
```

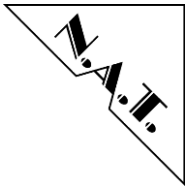
Parameter Description :

```
#:        number of PLL to configure
          1 - PLL #1
          2 - PLL #2

mode:     PLL operation mode
          1 - automatic selection between IC1 – IC8
          2 - forced mode (reference according to
              parameter ref)
          3 - free running mode

ref:      PLL reference input (only for forced mode)
          1 - PLL input IC1
          2 - PLL input IC2
          3 - PLL input IC3
          4 - PLL input IC4
          5 - PLL input IC5
          6 - PLL input IC6
          7 - PLL input IC7
          8 - PLL input IC8
```

Example:

```
clk_gen4_pll_mode  = 1, 1, 0
```

Configure the PLL 1 to automatically select the reference input between IC1 – IC8.



10.2.3.3 PLL OC Clock Output Configuration

Description:
Determines which frequency shall be output on a specific PLL clock output (OC).

Syntax:

```
clk_gen4_oc_cfg = oc, freq
```

Parameter Description:

```
oc:        output clock # of pll
           1 - PLL OC1
           2 - PLL OC2
           3 - PLL OC3
           4 - PLL OC4
           5 - PLL OC5
           6 - PLL OC6
           7 - PLL OC7

freq:      output clock frequency (in Hz)
```

Example:

```
clk_gen4_oc_cfg =  1, 8000
```

Outputs 8 kHz on PLL clock output OC1

10.2.3.4 Clock Output Configuration

Description:
Determines which clock source is driven on a specific clock output of the clock module.

Syntax :

```
clk_gen4_out = dst, src
```

Parameter Description :

```
dst: destination clock identifier
      1 - CLK1 AMC 1
      2 - CLK1 AMC 2
      3 - CLK1 AMC 3
      4 - CLK1 AMC 4
      5 - CLK1 AMC 5
      6 - CLK1 AMC 6
      7 - CLK1 AMC 7
      8 - CLK1 AMC 8
      9 - CLK1 AMC 9
     10 - CLK1 AMC 10
     11 - CLK1 AMC 11
     12 - CLK1 AMC 12
```

```
        13 - CLK2 AMC  1
        14 - CLK2 AMC  2
        15 - CLK2 AMC  3
        16 - CLK2 AMC  4
        17 - CLK2 AMC  5
        18 - CLK2 AMC  6
        19 - CLK2 AMC  7
        20 - CLK2 AMC  8
        21 - CLK2 AMC  9
        22 - CLK2 AMC 10
        23 - CLK2 AMC 11
        24 - CLK2 AMC 12
        25 - CLK3 AMC  1
        26 - CLK3 AMC  2
        27 - CLK3 AMC  3
        28 - CLK3 AMC  4
        29 - CLK3 AMC  5
        30 - CLK3 AMC  6
        31 - CLK3 AMC  7
        32 - CLK3 AMC  8
        33 - CLK3 AMC  9
        34 - CLK3 AMC 10
        35 - CLK3 AMC 11
        36 - CLK3 AMC 12
        37 - CLK1 Update
        38 - CLK3 Update
        39 - EXT single ended 1
        40 - EXT single ended 2
        41 - EXT single ended 3
        42 - EXT single ended 4
        43 - EXT differential 1
        44 - EXT differential 2
        45 - PLL IC1
        46 - PLL IC2
        47 - PLL IC3
        48 - PLL IC4
        49 - PLL IC5
        50 - PLL IC6
        51 - PLL IC7
        52 - PLL IC8
        53 - PLL SYNC1
        54 - PLL SYNC2
        55 - PLL SYNC3
        56 - SYNC CLK input

    src: source clock identifier
          0 - disabled
          1 - CLK1 AMC  1
          2 - CLK1 AMC  2
          3 - CLK1 AMC  3
```

```
 4 - CLK1 AMC  4
 5 - CLK1 AMC  5
 6 - CLK1 AMC  6
 7 - CLK1 AMC  7
 8 - CLK1 AMC  8
 9 - CLK1 AMC  9
10 - CLK1 AMC 10
11 - CLK1 AMC 11
12 - CLK1 AMC 12
13 - CLK2 AMC  1
14 - CLK2 AMC  2
15 - CLK2 AMC  3
16 - CLK2 AMC  4
17 - CLK2 AMC  5
18 - CLK2 AMC  6
19 - CLK2 AMC  7
20 - CLK2 AMC  8
21 - CLK2 AMC  9
22 - CLK2 AMC 10
23 - CLK2 AMC 11
24 - CLK2 AMC 12
25 - CLK3 AMC  1
26 - CLK3 AMC  2
27 - CLK3 AMC  3
28 - CLK3 AMC  4
29 - CLK3 AMC  5
30 - CLK3 AMC  6
31 - CLK3 AMC  7
32 - CLK3 AMC  8
33 - CLK3 AMC  9
34 - CLK3 AMC 10
35 - CLK3 AMC 11
36 - CLK3 AMC 12
37 - CLK1 Update
38 - CLK3 Update
39 - EXT single ended 1
40 - EXT single ended 2
41 - EXT single ended 3
42 - EXT single ended 4
43 - EXT differential 1
44 - EXT differential 2
45 - PLL OC1
46 - PLL OC2
47 - PLL OC3
48 - PLL OC4
49 - PLL OC5
50 - PLL OC6
51 - PLL OC7
52 - PLL FSYNC 8kHz
53 - PLL MFSYNC 2kHz
```

```
54 - 12.8MHz OSC
55 - 25MHz OSC (only with HCSL option)
56 - LEVEL 0
57 - LEVEL 1
58 - SYNC_CLK
```

NOTE: Refer to the Clock-Module Technical Reference Manual for a description of the different clock types.

Example:

```
clk_gen4_out = 1, 50
```

Output the PLL clock output OC6 on CLK 1 of AMC slot 1.

10.2.4 Clock Module Version Physics

10.2.4.1 Clock Output Configuration

Description:
Determines which clock source is driven on a specific clock output of the clock module.

Syntax:

```
clk_phys_out = dst, src
```

Parameter Description:

```
dst:  destination clock identifier
       1 - CLK1 AMC 1
       2 - CLK1 AMC 2
       3 - CLK1 AMC 3
       4 - CLK1 AMC 4
       5 - CLK1 AMC 5
       6 - CLK1 AMC 6
       7 - CLK1 AMC 7
       8 - CLK1 AMC 8
       9 - CLK1 AMC 9
      10 - CLK1 AMC 10
      11 - CLK1 AMC 11
      12 - CLK1 AMC 12
      13 - X
      14 - X
      15 - X
      16 - X
      17 - CLK2 AMC  1
      18 - CLK2 AMC  2
      19 - CLK2 AMC  3
      20 - CLK2 AMC  4
      21 - CLK2 AMC  5
      22 - CLK2 AMC  6
      23 - CLK2 AMC  7
      24 - CLK2 AMC  8
      25 - CLK2 AMC  9
      26 - CLK2 AMC 10
      27 - CLK2 AMC 11
      28 - CLK2 AMC 12
      29 - X
      30 - X
      31 - X
      32 - X
      33 - CLK3 AMC  1
      34 - CLK3 AMC  2
      35 - CLK3 AMC  3
      36 - CLK3 AMC  4
      37 - CLK3 AMC  5
      38 - CLK3 AMC  6
```

```
        39 - CLK3 AMC  7
        40 - CLK3 AMC  8
        41 - CLK3 AMC  9
        42 - CLK3 AMC 10
        43 - CLK3 AMC 11
        44 – CLK3 AMC 12

src:  source clock identifier
       0 - disabled
       1 - CLK1 AMC  1
       2 - CLK1 AMC  2
       3 - CLK1 AMC  3
       4 - CLK1 AMC  4
       5 - CLK1 AMC  5
       6 - CLK1 AMC  6
       7 - CLK1 AMC  7
       8 - CLK1 AMC  8
       9 - CLK1 AMC  9
      10 - CLK1 AMC 10
      11 - CLK1 AMC 11
      12 - CLK1 AMC 12
      13 - X
      14 - X
      15 - X
      16 - X
      17 - CLK2 AMC  1
      18 - CLK2 AMC  2
      19 - CLK2 AMC  3
      20 - CLK2 AMC  4
      21 - CLK2 AMC  5
      22 - CLK2 AMC  6
      23 - CLK2 AMC  7
      24 - CLK2 AMC  8
      25 - CLK2 AMC  9
      26 - CLK2 AMC 10
      27 - CLK2 AMC 11
      28 - CLK2 AMC 12
      35 - EXT single ended 1 (INPUT  SMA 1)
      41 - 100MHz OSC (only with HCSL option)
```

NOTE: Refer to the Clock-Module Technical Reference Manual for a description of the different clock types.

Example:

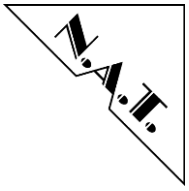
```
clk_phys_out = 1, 41
```

Output the 100MHz OSC clock output on CLK 1 of AMC slot 1.

The destination and source clock identifiers in the previous parameter description of the clock output configuration are considering a 1-to-1 connecting backplane. In case a NATIVE-R5 or NATIVE-C5 chassis is used the following clock identifier mapping have to be considered due to the backplane topology of these chassis.

dst	1-To-1 Connecting Backplane	NATIVE-C5 NATIVE-R5	comment
1	1 - CLK1 AMC 1	CLK1 AMC 2	
2	2 - CLK1 AMC 2	CLK1 AMC 3	
3	3 - CLK1 AMC 3	CLK1 AMC 4	
4	4 - CLK1 AMC 4	CLK1 AMC 5	
5	5 - CLK1 AMC 5	CLK1 AMC 6	
6	6 - CLK1 AMC 6	CLK1 AMC 7	
7	7 - CLK1 AMC 7	CLK1 AMC 1	
8	8 - CLK1 AMC 8		connection not available in Native-C5
9	9 - CLK1 AMC 9		connection not available in Native-C5
10	10 - CLK1 AMC 10		connection not available in Native-C5
11	11 - CLK1 AMC 11		connection not available in Native-C5
12	12 - CLK1 AMC 12		connection not available in Native-C5
17	17 - CLK2 AMC 1	CLK2 AMC 2	
18	18 - CLK2 AMC 2	CLK2 AMC 3	
19	19 - CLK2 AMC 3	CLK2 AMC 4	
20	20 - CLK2 AMC 4	CLK2 AMC 5	
21	21 - CLK2 AMC 5	CLK2 AMC 6	
22	22 - CLK2 AMC 6	CLK2 AMC 7	
23	23 - CLK2 AMC 7	CLK2 AMC 1	
24	24 - CLK2 AMC 8		connection not available in Native-C5
25	25 - CLK2 AMC 9		connection not available in Native-C5
26	26 - CLK2 AMC 10		connection not available in Native-C5
27	27 - CLK2 AMC 11		connection not available in Native-C5
28	28 - CLK2 AMC 12		connection not available in Native-C5
33	33 - CLK3 AMC 1	CLK3 AMC 2	
34	34 - CLK3 AMC 2	CLK3 AMC 3	
35	35 - CLK3 AMC 3	CLK3 AMC 4	
36	36 - CLK3 AMC 4	CLK3 AMC 5	
37	37 - CLK3 AMC 5	CLK3 AMC 6	
38	38 - CLK3 AMC 6	CLK3 AMC 7	
39	39 - CLK3 AMC 7	CLK3 AMC 1	
40	40 - CLK3 AMC 8		connection not available in Native-C5
41	41 - CLK3 AMC 9		connection not available in Native-C5
42	42 - CLK3 AMC 10		connection not available in Native-C5
43	43 - CLK3 AMC 11		connection not available in Native-C5
44	44 - CLK3 AMC 12		connection not available in Native-C5

Table 10-1 Clock Output Identifier Mapping in NATIVE-C5 or NATIVE-R5 chassis



10.2.4.2 Clock Termination Configuration

Description:
Determines if 100 R termination in the multiplexer is enabled or disabled.

Syntax:

```
clk_phys_termination = dst, termination
```

Parameter Description:

```
dst:        clock identifier
             1 - CLK1 AMC  1
             2 - CLK1 AMC  2
             3 - CLK1 AMC  3
             4 - CLK1 AMC  4
             5 - CLK1 AMC  5
             6 - CLK1 AMC  6
             7 - CLK1 AMC  7
             8 - CLK1 AMC  8
             9 - CLK1 AMC  9
            10 - CLK1 AMC 10
            11 - CLK1 AMC 11
            12 - CLK1 AMC 12
            13 - X
            14 - X
            15 - X
            16 - X
            17 - CLK2 AMC  1
            18 - CLK2 AMC  2
            19 - CLK2 AMC  3
            20 - CLK2 AMC  4
            21 - CLK2 AMC  5
            22 - CLK2 AMC  6
            23 - CLK2 AMC  7
            24 - CLK2 AMC  8
            25 - CLK2 AMC  9
            26 - CLK2 AMC 10
            27 - CLK2 AMC 11
            28 - CLK2 AMC 12
            29 - X
            30 - X
            31 - X
            32 - X

termination: enable/disable 100 R termination in multiplexer
```

Example:

```
clk_phys_termination =  28, 1
```

Enables 100 R termination on CLK 2 of AMC slot 12.

10.2.5 Clock Module Version USB

10.2.5.1 Clock Output Configuration

Description:
Determines if the clock source 100MHz OSC is driven on a specific clock output of the clock module.

Syntax:

```
clk_usb_out = dst, src
```

Parameter Description:

```
dst:  destination clock identifier
       1 - CLK3 AMC 1
       2 - CLK3 AMC 2
       3 - CLK3 AMC 3
       4 - CLK3 AMC 4
       5 - CLK3 AMC 5
       6 - CLK3 AMC 6
       7 - CLK3 AMC 7
       8 - CLK3 AMC 8
       9 - CLK3 AMC 9
      10 - CLK3 AMC 10
      11 - CLK3 AMC 11
      12 - CLK3 AMC 12

src:  source clock identifier
       0 - disabled
       1 - 100MHz OSC (only with HCSL option)
```

NOTE: Refer to the Clock-Module Technical Reference Manual for a description of the different clock types.

Example:

```
clk_usb_out = 6, 1
```

Output the 100MHz OSC clock output on CLK 3 of AMC slot 6.

10.2.6 Common Commands for all Clock Modules

10.2.6.1 Write Clock Module Register

Description:
Write a value to a clock module register.

NOTE: This configuration item only should be used if the required functionality has not yet been implemented by another configuration item. Please ask NAT for adding support of the needed functionality, as the register layout might differ depending on the clock module version used!

Syntax:

```
clk_wreg = offs, value
```

Parameter Description:

```
offs:     register offset

value:    value to write
```

Example:

```
clk_wreg = 0x10, 0xff
```

Write value 0xff to register at offset 0x10.

10.2.6.2 OR Value to Clock Module Register

Description:
ORs a value to a clock module register. This can be used to set specific bits of a register.

NOTE: This configuration item only should be used if the required functionality has not yet been implemented by another configuration item. Please ask NAT for adding support of the needed functionality, as the register layout might differ depending on the clock module version used!

Syntax:

```
clk_orreg = offs, value
```

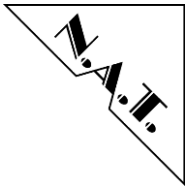
Parameter Description:

```
offs:     register offset

value:    value to OR with current register value
```

Example:

```
clk_orreg = 0x10, 0x01
```

Set bit 0 (0x01) in clock module register at offset 0x10.



10.2.6.3 AND Value to Clock Module Register

Description:
ANDs a value to a clock module register. This can be used to clear specific bits of a register.

NOTE: This configuration item only should be used if the required functionality has not yet been implemented by another configuration item. Please ask NAT for adding support of the needed functionality, as the register layout might differ depending on the clock module version used!

Syntax:

```
clk_andreg = offs, value
```

Parameter Description:

```
offs:     register offset

value:    value to AND with current register value
```

Example:

```
clk_andreg = 0x10, 0x7f
```

Clear bit 7 (0x80) in clock module register at offset 0x10.

10.3 SRIO Module Configuration

The SRIO module mounted on the NAT-MCH can be configured via a text based script file, similar to the configuration file used for the Ethernet switch on the NAT-MCH.

The different functions of the SRIO module can be set by using so called "configuration items". Each configuration item consists of an identifier and one or more parameters.

Comments within the configuration begin with a hash character ('#'), like:

```
#
# Item <<srio_port_init>>: initialize SRIO port
#
```

The current SRIO module configuration can be downloaded from the NAT-MCH e.g. by using the "Script Management" function in the web based configuration interface (refer to chapter 9). The generated text file can be changed with a standard text editor and can be used as a starting point for a user defined configuration. The changed configuration can be uploaded again, e.g. via the web based configuration interface.

The following chapters describe the configuration items which can be used to configure the SRIO module. For information about the functionality provided by the SRIO module please refer to the latest SRIO-Module Technical Reference Manual.

10.3.1 SRIO Port Initialization

Description:
Initializes a SRIO switch port and configures it to a specific port speed. Please refer to the "Switch to Fabric Port Mapping" tables in the SRIO Technical Reference Manuals.

Syntax:

```
srio_port_init = switch_dev, port, speed
```

Parameter Description:

```
switch_dev:    switch device number 0,1
port:          physical switch port number
speed:         port speed
               0 - 1.25  Gbaud
               1 - 2.5   Gbaud
               2 - 3.125 Gbaud
               3 - 5.0   Gbaud
               4 - 6.25  Gbaud
```

Example:

```
srio_port_init = 0, 9, 2
```

Initializes port 9 of the first switch device to 3.125 Gbaud.

10.3.2 Write SRIO Module Byte Register

Description:
Write a value to a SRIO module byte register.

Syntax:

```
srio_wregb = offs, value
```

Parameter Description:

```
offs:     register offset

value:    value to write
```

Example:

```
srio_wregb = 0x10, 0xff
```

Write value 0xff to register at offset 0x10.

10.3.3 OR Value to SRIO Module Byte Register

Description:
ORs a value to a SRIO module byte register. This can be used to set specific bits of a register.

Syntax:

```
srio_orregb = offs, value
```

Parameter Description:

```
offs:     register offset

value:    value to OR with current register value
```

Example:

```
srio_orregb = 0x10, 0x01
```

Set bit 0 (0x01) in SRIO module byte register at offset 0x10.

10.3.4 AND Value to SRIO Module Register

Description:
ANDs a value to a SRIO module byte register. This can be used to clear specific bits of a register.

Syntax:

```
srio_andregb = offs, value
```

Parameter Description:

```
offs:     register offset

value:    value to AND with current register value
```

Example:

```
srio_andregb = 0x10, 0x7f
```

Clear bit 7 (0x80) in SRIO module register at offset 0x10.

10.3.5 Write SRIO Module Switch Register (Indirect Long)

Description:
Write a value to a SRIO module switch register. Please refer to the "Switch to Fabric Port Mapping" tables in the SRIO Technical Reference Manuals to select correct switch_dev parameter. Please refer to the SRIO switch User Manual for a description of the switch registers.

Syntax:

```
srio_wregil = switch_dev, offs, value
```

Parameter Description:

```
switch_dev:    switch device number 0,1

offs:          register offset

value:         value to write
```

Example:

```
srio_wregil = 0x10, 0xff
```

Write value 0xff to SRIO module switch register at offset 0x10.

10.3.6 OR Value to SRIO Module Switch Register (Indirect Long)

Description:
ORs a value to a SRIO module switch register. This can be used to set specific bits of a register. Please refer to the "Switch to Fabric Port Mapping" tables in the SRIO Technical Reference Manuals to select correct switch_dev parameter. Please refer to the SRIO switch User Manual for a description of the switch registers.

Syntax:

```
srio_orregil = switch_dev, offs, value
```

Parameter Description:

```
switch_dev:    switch device number 0,1

offs:     register offset

value:    value to OR with current register value
```

Example:

```
srio_orregb = 0x10, 0x01
```

Set bit 0 (0x01) in SRIO module switch register at offset 0x10.

10.3.7 AND Value to SRIO Module Switch Register (Indirect Long)

Description:
ANDs a value to a SRIO module switch register. This can be used to clear specific bits of a register. Please refer to the "Switch to Fabric Port Mapping" tables in the SRIO Technical Reference Manuals to select correct switch_dev parameter. Please refer to the SRIO switch User Manual for a description of the switch registers.

Syntax:

```
srio_andregil = switch_dev, offs, value
```

Parameter Description:

```
switch_dev:    switch device number 0,1

offs:     register offset

value:    value to AND with current register value
```

Example:

```
srio_andregil = 0x10, 0x7f
```

Clear bit 7 (0x80) in SRIO module switch register at offset 0x10.

10.3.8 SET port route table entry on SRIO module V2.x

Description:
SETs port route table entry on SRIO2 module switch. This configuration parameter can be used for setting up the route table during ekeying only. Maximum 16 IDs per port can be configured manually via this parameter! Please refer to the "Switch to Fabric Port Mapping" tables in the SRIO Technical Reference Manuals to select correct switch_dev parameter. Please refer to the SRIO switch User Manual for a description of the switch registers.

Syntax:

```
srio_set_port_route = switch_dev, destid, port
```

Parameter Description:

```
switch_dev:    switch device number 0,1

destid: destination ID

port: port number 0, 1,..., 11
```

Example:

```
srio_set_port_route = 0, 1, 9
```

Configures switch 0 with a routing entry of ID#1 is related to port 9.

10.3.9 SET Port Link Timeout CSR[TIMEOUT] of SRIO module V2.x

Description:
SETs value TIMEOUT in the switch register Port Link Timeout Control CSR. The reset value of TIMEOUT in this switch register is 0xFFFFFF which represents a timeout period of TIMEOUT*352ns = 5,9sec. According to the User Manual of the switch this value shall be changed to a value of 20-50us. In case that the SRIO2 script configuration is not enabled the MCH firmware initializes the value TIMEOUT to 0x8E (0x8E * 352ns = 50us). **If the SRIO2 script configuration is enabled (**SRIO module configuration source**) this configuration parameter has to be determined to a value which represents a timeout period of 20-50us.** Please refer to the "Switch to Fabric Port Mapping" tables in the SRIO Technical Reference Manuals to select correct switch_dev parameter. Please refer to the SRIO switch User Manual for a description of the switch registers.

Syntax:

```
srio_set_port_lnk_timeout = switch_dev, timeout
```

Parameter Description:

```
switch_dev:    switch device number 0,1

timeout: t(Timeout)/352ns
```

Example:

```
srio_set_port_lnk_timeout = 0, 57
```

Configures switch 0 with a timeout period of 20us (57 * 352ns = 20us).

10.3.10 SRIO Fru Max Speed

Description:

This configuration command reduces the maximum SRIO speed of a switch port that is connected to the related FRU device ID during e-keying. This configuration parameter decreases the maximum physical link speed as determined by the switch type hardware. During E-Keying process this value is compared against the value of LinkTypeExt field of the related FRU device ID Link Descriptor and the minimum of both values is agreed as the link speed. The speed can only be decreased against its maximum physical possible value but not increased.

Syntax:

```
srio_fru_max_speed = fru_id, speed
```

Parameter Description:

```
fru_id:   FRU device Id of related AMC that connects to a
          switch port (AMC1…12, switch interconnects and Face
          plate uplinks; refer to Table 10-2: SRIO Switch to FRU device
          ID Mapping for details)
speed:    port speed
          0 - 1.25  Gbaud
          1 - 2.5   Gbaud
          2 - 3.125 Gbaud
          3 - 5.0   Gbaud
          4 - 6.25  Gbaud

Note:     Both face plate uplinks can only work in a proper way
          with a reduced max. speed of 5.0 GBaud!
```

Example:

```
srio_fru_max_speed = 5, 2
```

The above example reduces the maximum SRIO speed of the switch port that connects to FRU device ID 5 (AMC1) to 3.125 Gbaud during e-keying.

Table 10-2: SRIO Switch to FRU device ID Mapping

# AMC Slot Fabric D-G	# FRU device ID
AMC1	5
AMC2	6
AMC3	7
AMC4	8
AMC5	9
AMC6	10
AMC7	11
AMC8	12
AMC9	13
AMC10	14
AMC11	15
AMC12	16
MCH Update (backplane connection to 2nd MCH)	240
1st switch interconnect	241
2nd switch interconnection	242
3rd switch interconnection	243
Face plate uplink 1	250
Face plate uplink 2	251

10.3.11 SRIO Fru NEG1_TAP

Description:
This configuration command configures the SRIO NEG1_TAP parameter of the selected SRIO lane of a switch port that connects to a FRU device ID to the specific value.

Syntax:

```
srio_fru_neg1_tap = fru_id, lane, neg1_tap_value
```

Parameter Description:

```
fru_id:          FRU device Id of related AMC switch lanes
                 (AMC1…12, switch interconnects and Face plate
                 uplinks; refer to Table 10-2: SRIO Switch to FRU
                 device ID Mapping for details)
lane:            lane number of the related switch port
                 0 - 1st lane of the selected port
                 1 - 2nd lane of the selected port
                 2 - 3rd lane of the selected port
                 3 - 4th lane of the selected port
                 4 - all four lanes of the selected port
neg1_tap_value: NEG1_TAP value
                 5 bit value (0-31)
```

Example:

```
srio_fru_neg1_tap = 5, 4, 2
```

Set the SRIO NEG1_TAP parameter of all related switch lanes of the FRU device ID 5 (AMC1) to 2.

10.3.12 SRIO Fru POS1_TAP

Description:
This configuration command configures the SRIO POS1_TAP parameter of the selected SRIO lane of a switch port that connects to a FRU device ID to the specific value.

Syntax:

```
srio_fru_pos1_tap = fru_id, lane, pos1_tap_value
```

Parameter Description:

```
fru_id:          FRU device Id of related AMC switch lanes
                 (AMC1…12, switch interconnects and Face plate
                 uplinks; refer to Table 10-2: SRIO Switch to FRU
                 device ID Mapping for details)
lane:            lane number of the related switch port
                 0 - 1st lane of the selected port
                 1 - 2nd lane of the selected port
                 2 - 3rd lane of the selected port
                 3 - 4th lane of the selected port
                 4 - all four lanes of the selected port
pos1_tap_value:  POS1_TAP value
                 6 bit value (0-63)
```

Example:

```
srio_fru_pos1_tap = 5, 4, 2
```

Set the SRIO POS1_TAP parameter of all related switch lanes of the FRU device ID 5 (AMC1) to 2.

10.4 Unmanaged AMC Module Configuration

Unmanaged AMC modules can be configured via a text based script file, similar to the configuration file used for the Ethernet switch or clock module on the NAT-MCH.

The payload power of an unmanaged AMC module can be activated by using a so called "configuration item". The configuration item consists of an identifier and three parameters. The following example shows the configuration item "amc_pwr_on" which can be used to activate the payload power of an unmanaged AMC module.

```
amc_pwr_on = 7, 20, 0
```

As shown above the configuration item identifier is always separated from the parameter list by an equals sign ('='). The parameters are always separated by a comma sign.

Comments within the configuration begin with a hash character ('#'), like:

```
#
# Item <<amc_pwr_on>>: initialize AMC pwr_on
#
# Syntax: amc_pwr_on = fru_id, current_max, t_delay
#
# Params: fru_id: amc fru id 5...30
#         current_max: maximum current consumption in 100mA
#                      steps
#         t_delay: pwr_on delay time in 1sec steps (max. 65535
#                  sec)
#
```

Example:

```
#
# Enable AMC 1 (fru id 5) with a maximum power consumption of
# 2.0A without a delay time
#
amc_pwr_on = 5, 20, 0
#
# Enable AMC 4 (fru id 8) with a maximum power consumption of
# 3.5A and a delay time of 35 seconds
#
amc_pwr_on = 8, 35, 35
```

The current AMC module configuration can be downloaded from the NAT-MCH e.g. by using the "Script Management" function in the web based configuration interface (refer to chapter 9). The generated text file can be changed with a standard text editor and can be used

as a starting point for a user defined configuration. The changed configuration can be uploaded again, e.g. via the web based configuration interface.

The AMC module configuration can also be stored in the onboard FLASH memory, e.g. by using the "Save current configuration to onboard FLASH memory" function in the web based configuration interface. This can be used to load the configuration for the AMC module during startup of the NAT-MCH.

Once the AMC module configuration is stored in the onboard FLASH memory an AMC module (PS1 signal necessary) is powered when it is inserted according to the configuration parameters. A power up reset in a system with an inserted module will also power up the AMC module according to the configuration parameters.

10.5 Watchdog Timer configuration

When the Watchdog Timer is enabled in the MCH settings it can be configured in more detail via the script file. Below you see the default configuration and the explanation of the different commands:

```
# Item <<wd_exception_reset>>: enable reset when an exception
#in NAT-MCH firmware occurs
#
# Syntax: wd_exception_reset = value
#
# Params: value: = 0 disabled; > 0 enabled
#
wd_exception_reset = 1

# Item <<wd_task_lost_reset>>: enable reset when one important
#task of NAT-MCH firmware is lost
#
# Syntax: wd_task_lost_reset = value
#
# Params: value: = 0 disabled; > 0 enabled
#
wd_task_lost_reset = 1

# Item <<wd_shutdown_after_reset>>: enable system shutdown
#after reset
#
# Syntax: wd_shutdown_after_reset = value
#
# Params: value: = 0 disabled; > 0 enabled
#
wd_shutdown_after_reset = 1
```

10.6 PCIe Module Configuration

The configuration of the HUB-PCIe-x48 (V2.x) and the HUB-PCIe-x80 module can be changed using the text based configuration script.
The following chapters describe the configuration items which can be used to configure the HUB-PCIe module via script. Note that the configuration items might differ depending on the HUB-PCIe module version.
In case that the configuration parameters of the PCIe Module Configuration have been modified by updating the flash script file a Power down and Power up cycle of the system has to be performed.

10.6.1 PCIe Virtual Switch Initialization

Description:

10.7 Initialize a virtual switch configuration of the PCIe switch. This configuration item provides the configuration option similar to the configuration option via the webserver interface "PCIe-Hub Module (optional)

PCIe Virtual Switch Configuration (optional)".

Syntax:

```
pcie_vs_cfg = vs_id, up_amc, nt_up_amc, ds_amc_list
```

Parameter Description:

```
vs_id:          Virtual Switch Id (VS_0, VS_1,...VS_5)
up_amc:         Upstream PCIe device name
nt_up_amc:      NT-Upstream PCIe device name (only for VS_0 for
                all others only NONE) or NONE for none
ds_amc_list:    Downstream PCIe device name
```

The possibility of choosing one of the following PCIe device naming is depending of the HUB-PCIe module and the backplane connection that are used in the system!

```
NONE    - none
AMC1_4  - AMC1,  port 4..7
AMC2_4  - AMC2,  port 4..7
AMC3_4  - AMC3,  port 4..7
AMC4_4  - AMC4,  port 4..7
AMC5_4  - AMC5,  port 4..7
AMC6_4  - AMC6,  port 4..7
AMC7_4  - AMC7,  port 4..7
AMC8_4  - AMC8,  port 4..7
```

```
AMC9_4   - AMC9,  port 4..7
AMC10_4  - AMC10, port 4..7
AMC11_4  - AMC11, port 4..7
AMC12_4  - AMC12, port 4..7
RTM_x16  - RTM,   only with HUB-PCIe-x80
OPT1     - OPT1,  only with HUB-PCIe-x80 (assembly
           option)
OPT2     - OPT2,  only with HUB-PCIe-x80 (assembly
           option)
AMC1_8   - AMC1,  port 8..11
AMC2_8   - AMC2,  port 8..11
AMC3_8   - AMC3,  port 8..11
AMC4_8   - AMC4,  port 8..11
AMC5_8   - AMC5,  port 8..11
AMC6_8   - AMC6,  port 8..11
AMC7_8   - AMC7,  port 8..11
AMC8_8   - AMC8,  port 8..11
AMC9_8   - AMC9,  port 8..11
AMC10_8  - AMC10, port 8..11
AMC11_8  - AMC11, port 8..11
AMC12_8  - AMC12, port 8..11
```

Example:

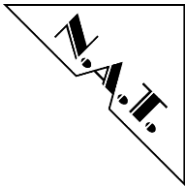
```
pcie_vs_cfg = VS_0, AMC1_4, NONE, AMC3_4, AMC5_4, AMC7_4
pcie_vs_cfg = VS_1, AMC2_4, NONE, AMC4_4, AMC6_4
```

This example initializes two virtual switches VS_0 and VS_1 as follow:

PCIe Virtual Switch configuration

Select Host AMCs (Upstream) for each virtual switch that shall be enabled first.
Select Host AMCs (Non-Transparent Upstream) for each virtual switch that shall be enabled afterwards.
Select which AMCs shall be connected to each virtual switch as downstream in the end.

Virtual Switch	Upstream AMC	NT-Upstream AMC	AMC 1 4..7	AMC 2 4..7	AMC 3 4..7	AMC 4 4..7	AMC 5 4..7	AMC 6 4..7	AMC 7 4..7
none			○	○	○	○	○	○	○
Virtual Switch 0	AMC 1_4	- none -	◉	○	◉	○	◉	○	◉
Virtual Switch 1	AMC 2_4		○	◉	○	◉	○	◉	○
Virtual Switch 2	- none -		○	○	○	○	○	○	○
Virtual Switch 3	- none -		○	○	○	○	○	○	○
Virtual Switch 4	- none -		○	○	○	○	○	○	○
Virtual Switch 5	- none -		○	○	○	○	○	○	○
Max. Link Speed			5.0 GT/s	2.5 GT/s	8.0 GT/s	5.0 GT/s	2.5 GT/s	8.0 GT/s	5.0 GT/s



10.7.1 PCIe Maximum Link Speed

Description:

10.8 Set the maximum link speed of an AMC related to a PCIe virtual switch port (HUB-PCIe-x48 PCB >= V2.0 or HUB-PCIe-x80 only). This configuration item provides the configuration option similar to the configuration option via the webserver interface "PCIe-Hub Module (optional)

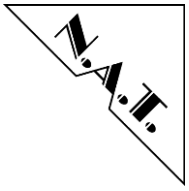
PCIe Virtual Switch Configuration (optional)".

Syntax:

```
pcie_lnk_speed_max = amc, speed
```

Parameter Description:

```
amc:    related AMC port

          AMC1_4  - AMC1,  port 4..7
          AMC2_4  - AMC2,  port 4..7
          AMC3_4  - AMC3,  port 4..7
          AMC4_4  - AMC4,  port 4..7
          AMC5_4  - AMC5,  port 4..7
          AMC6_4  - AMC6,  port 4..7
          AMC7_4  - AMC7,  port 4..7
          AMC8_4  - AMC8,  port 4..7
          AMC9_4  - AMC9,  port 4..7
          AMC10_4 - AMC10, port 4..7
          AMC11_4 - AMC11, port 4..7
          AMC12_4 - AMC12, port 4..7
          RTM_x16 - RTM,   only with HUB-PCIe-x80
          OPT1    - OPT1,  only with HUB-PCIe-x80 (assembly
                     option)
          OPT2    - OPT2,  only with HUB-PCIe-x80 (assembly
                     option)
          AMC1_8  - AMC1,  port 8..11
          AMC2_8  - AMC2,  port 8..11
          AMC3_8  - AMC3,  port 8..11
          AMC4_8  - AMC4,  port 8..11
          AMC5_8  - AMC5,  port 8..11
          AMC6_8  - AMC6,  port 8..11
          AMC7_8  - AMC7,  port 8..11
          AMC8_8  - AMC8,  port 8..11
          AMC9_8  - AMC9,  port 8..11
          AMC10_8 - AMC10, port 8..11
          AMC11_8 - AMC11, port 8..11
```

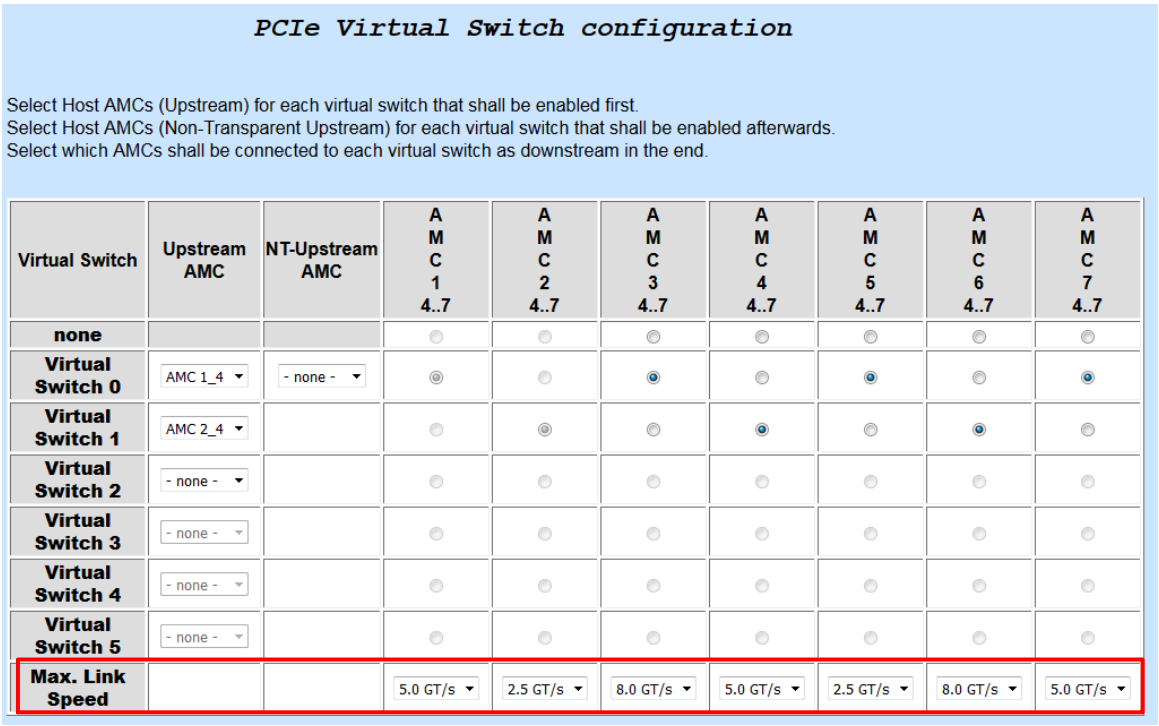


```
            AMC12_8 - AMC12, port 8..11

speed:  max. link speed
            LS_2_5GTs - 2.5   GT/s
            LS_5_0GTs - 5.0   GT/s
            LS_8_0GTs - 8.0   GT/s
```

Example:

```
pcie_lnk_speed_max = AMC1_4, LS_5_0GTs
pcie_lnk_speed_max = AMC2_4, LS_2_5GTs
pcie_lnk_speed_max = AMC4_4, LS_5_0GTs
pcie_lnk_speed_max = AMC5_4, LS_2_5GTs
pcie_lnk_speed_max = AMC7_4, LS_5_0GTs
```

This example initializes the maximum link speed of the AMC as follow:



10.8.1 PCIe Upstream slot power up delay

Description:
Set the parameter <Upstream slot power up delay> of the MCH configuration(HUB-PCIe-x48 PCB >= V2.0 or HUB-PCIe-x80 only). This configuration item provides the configuration option similar to the configuration option via the MCH configuration interface "5.4.7 PCIe Switch Configuration [9] (optional)".

Syntax:

```
pcie_up_delay = value
```

Parameter Description:

```
value: upstream slot power up delay time in seconds (min. 5sec)
```

Example:

```
pcie_up_delay = 10
```

This example initializes the parameter <Upstream slot power up delay> to 10 seconds.

10.8.2 PCIe hot plug delay for AMCs

Description:
Set the parameter <PCIe hot plug delay for AMCs> of the MCH configuration. This configuration item provides the configuration option similar to the configuration option via the MCH configuration interface "5.4.7 PCIe Switch Configuration [9] (optional)".

Syntax:

```
pcie_amc_up_delay = value
```

Parameter Description:

```
value: PCIe hot plug delay time for AMCs in seconds
       (default 0sec)
```

Example:

```
pcie_amc_up_delay = 2
```

This example initializes the parameter <PCIe hot plug delay for AMCs> to 2 seconds.

10.8.3 PCIe 100 MHz spread spectrum clock

Description:
Set the parameter <100 MHz spread spectrum clock> of the MCH configuration. This configuration item provides the configuration option similar to the configuration option via the MCH configuration interface "5.4.7 PCIe Switch Configuration [9] (optional)".

Syntax:

```
pcie_spread = value
```

Parameter Description:

```
value: 0 = disabled (default); 1 = enabled

Note: SSC only selectable when no LOSC option assembled
      and HUB-PCIe-x48 PCB >= V2.0 or HUB-PCIe-x80!
```

Example:

```
pcie_spread = 1
```

This example initializes the parameter <100 MHz spread spectrum clock> to 1 and enables the 100 MHz spread spectrum clock.

10.8.4 PCIe hot plug support

Description:
Set the parameter <hot plug support> of the MCH configuration. This configuration item provides the configuration option similar to the configuration option via the MCH configuration interface "5.4.7 PCIe Switch Configuration [9] (optional)".

Syntax:

```
pcie_hot_plug = value
```

Parameter Description:

```
value: 0 = disabled (default); 1 = enabled
```

Example:

```
pcie_hot_plug = 1
```

This example initializes the parameter <hot plug support> to 1 and enables the PCIe Hot Plug Support.

10.8.5 PCIe early ekey (before payload)

Description:
Set the parameter <PCIe early ekey (before payload)> of the MCH configuration. This configuration item provides the configuration option similar to the configuration option via the MCH configuration interface "5.4.7 PCIe Switch Configuration [9] (optional)".

Syntax:

```
pcie_ek_early = value
```

Parameter Description:

```
value: 0 = disabled (default); 1 = enabled
```

Example:

```
pcie_ek_early = 1
```

This example initializes the parameter <PCIe early ekey (before payload)> to 1 and enables the PCIe early ekey (before payload).

10.8.6 PCIe SSC isolation for RTM

Description:
Set the parameter <SSC isolation for RTM> of the MCH configuration (HUB-PCIe-x80 only). This configuration item provides the configuration option similar to the configuration option via the MCH configuration interface "5.4.7 PCIe Switch Configuration [9] (optional)".

Syntax:

```
pcie_ssc_iso_rtm = value
```

Parameter Description:

```
value: 0 = disabled (default); 1 = enabled
```

```
Note: Configuration item only available on a HUB-PCIe-x80!
```

Example:

```
pcie_ssc_iso_rtm = 1
```

This example initializes the parameter <SSC isolation for RTM> to 1 and enables the SSC isolation for RTM.

10.8.7 PCIe allow reset propagation for RTM

Description:
Set the parameter <allow reset propagation> of the MCH configuration (HUB-PCIe-x80 only). This configuration item provides the configuration option similar to the configuration option via the MCH configuration interface "5.4.7 PCIe Switch Configuration [9] (optional)".

Syntax:

```
pcie_res_from_rtm = value
```

Parameter Description:

```
value: 0 = disabled (default); 1 = enabled
```

```
Note: Configuration item only available on a HUB-PCIe-x80!
```

Example:

```
pcie_res_from_rtm = 1
```

This example initializes the parameter <allow reset propagation> to 1 and enables the reset propagation for RTM.

10.9 Configure primary/secondary MCH with the same script file

In a system with two MCHs (a redundant system) it is helpful to setup both MCHs in one script file and load this up to both MCHs.
To distinguish each MCH and its modules (PCIe/SRIO/XAUI/Clock) there are three parameters before every part of the script file:

Syntax:
```
mch_id = value1
mez_id = value2
inst_id = value3
```

Parameter Description:
```
value1: 0 = MCH site 1; 1 = MCH site 2
value2: 0 = base module; 1 = clock module; 2 = hub module
value3: 0 = default; (unused up to now)
```

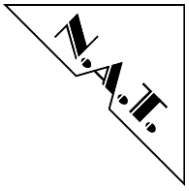
Example:
```
mch_id = 0
mez_id = 2
inst_id = 0
« other script parameter »
```

This example configures the hub module for MCH 1.

To add the configuration for the hub module of the second MCH just duplicate all these parameter including `« other script parameter »` and change the "mch_id" parameter to "1".

Example:
```
mch_id = 1
mez_id = 2
inst_id = 0
« other script parameter »
```

Note:
- You have to change the mch_id for all parts of the script
- You have to upload the script file to both MCHs, because an MCH can only configure itself.
- Base module and hub module Ethernet configuration on both MCHs are not supported up to now (will be added in a future release).

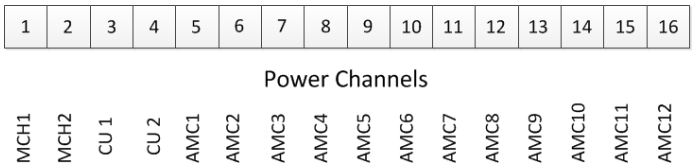


11 N+1 Power Module Configurations

Since the MCH firmware version 2.15 the MCH supports up to four power modules. With three or four power modules it is possible to realize a N+1 redundancy or redundant load sharing configuration. The MCH gets the information how to configure the power modules from the backplane FRU. To configure the backplane FRU you can use NATView.

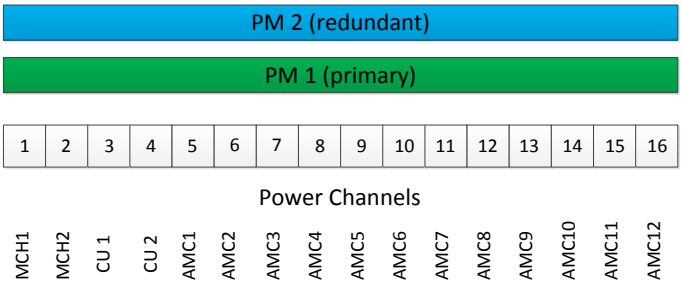
11.1 Power Channel

In an uTCA system there are up to16 power channels. There is one power channel for every FRU in the system (MCHs, CUs and AMCs). To every power channel up to two PMs can be assigned – one primary and one secondary. The information about the assignment of the PMs to the power channels is stored in the backplane FRU information.



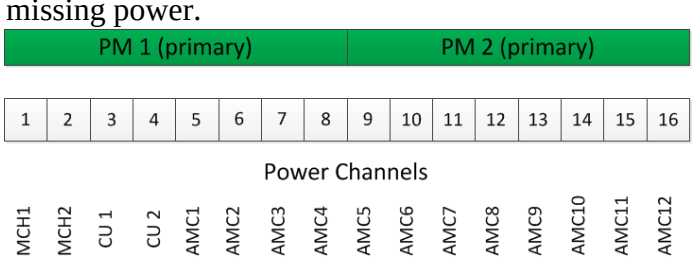
11.2 Redundancy (2 PMs)

In a traditional redundant PM setup there are two PMs. Every PM is assigned to every power channel. When the primary PM fails the secondary PM will take over and the system will work further without interruption. In a redundancy setup you can use only the power of one PM.



11.3 Load Sharing (2 PMs)

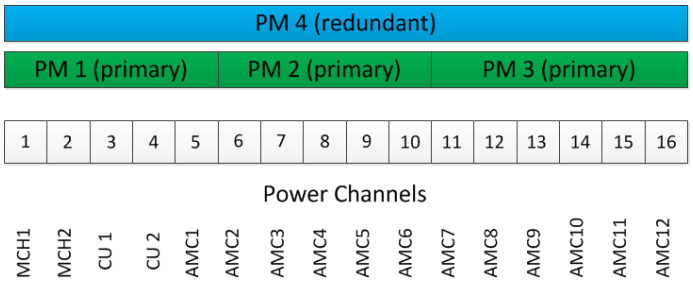
In a load sharing configuration there is only one PM assigned to a power channel. The total power of all PMs can be used for the system, but when a PM fails every FRU assigned to this PM fails also due to missing power.



11.4 N+1 Redundancy (3/4 PMs)

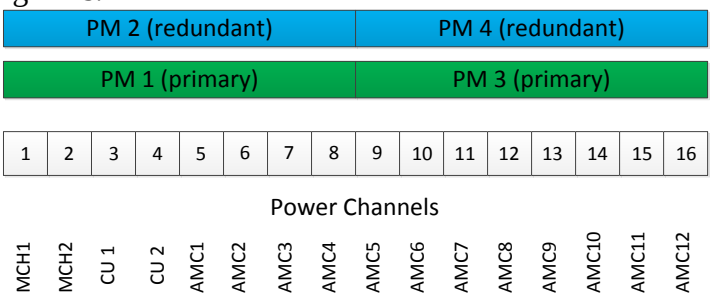
In a N+1 redundant configuration you have two or three primary PMs and one secondary PM. The power budget for the complete system is the sum of two or three primary PMs.

When one primary PM fails the redundant PM will take over for this PM immediately. If a second PM fails in addition, the power budget of the redundant PM can become the limitation.



11.5 Redundant Load Sharing (4 PMs)

In a redundant load sharing configuration there are two primary and two secondary PMs. The power budget for the complete system is the sum of two PMs. The system will still works with up to two failing PMs.



12 Hardware Releases

Version	Status
V1.0	Early access boards, for evaluation purpose only
V1.1	Pre-Series boards
V2.x	Series boards with Harting-Plug

Table 12-1 PCB Releases

12.1 Known Hardware Issues

V2.x PCB release: there are no hardware issues known at the time this manual was released.
V2.0 PCB release: there are no hardware issues known at the time this manual was released.

- Boards with release codes earlier than 070717 need to be upgraded to support MCH hot swap capability and MCH redundancy support

V1.1 PCB release:

- Boards with release codes earlier than 070717 need to be upgraded to support MCH hot swap capability and MCH redundancy support.

V1.0 PCB release:
2 Hot swap support not possible

13 Give us a Hint !

This MCH is a quite new product based on a new technology and it has to work together with products from other vendors which might be new as well. If you discover a problem, you can help us improving the product and fixing the problem quickly. The MCH configuration menu allows you to set debug options for certain functionalities (see chapter 5 and chapter 8)
In case you discover a problem please switch on the respective debug option and provide us with a log file taken from the output of the console port together with a brief description of the problem – we will care.
Thank you in advance!

N.A.T. GmbH

Appendix A Connector Pinouts

Appendix A 1 Console Connector – RS232

The following table shows the pin assignment of the signals of the RS232 interface, which is wired to the Mini-USB connector S1.

The parameters of the console port are: ***19200, 8, N, 1***

Pin No.	Signal	Signal	Pin No.
1	PSC0 RTS	PSC0 RXD	2
3	PSC0 TXD	PSC0 CTS	4
5	GND		

Table 13-1 Pin Assignment of the Front-panel Connector S1 (RS232)

Appendix A 2 Ethernet Connector – GbE Uplink

The following table shows the pin assignment of the RJ45 connector of the GbE Uplink port. This connector carries the 1000BaseT signals of the Ethernet interface of the Gigabit Ethernet Switch.

Pin No.	Signal	Signal	Pin No.
1	MDI0+	MDI0-	2
3	MDI1+	MDI2+	4
5	MDI2-	MDI1-	6
7	MDI3+	MDI3-	8

Table 13-2 Pin Assignment of the Front-panel Connector S2 (GbE Uplink)

Appendix A 3 Ethernet Connector – 100 BaseT Management Port

The following table shows the pin assignment of the RJ45 connector that carries the 100BaseT signals of the Ethernet interface of the ColdFire CPU. Termination is the 100BaseT termination used for pins 4, 5, 7, and 8.

Pin No.	Signal	Signal	Pin No.
1	TX+	TX-	2
3	RX+	Term.	4
5	Term.	RX-	6
7	Term.	Term.	8

Table 13-3 Pin Assignment of the Front-panel Connector (100 BaseT)

Appendix A 4 External Clock Reference Connector

The following table shows the pin assignment of the signals of the external reference clock interface.
Gen3 only: SMA1 is located directly besides the USB console jack, SMA2 is placed next to GbE2 interface.

Pin No.	Signal	Signal	Pin No.
Center	EXTREF_P	EXTREF_N	Shield

Table 13-4 Pin Assignment of the Clock Connector S3

Appendix A 5 NAT-MCH Base6/12 – Connector Tongue 1

Pin No.	MCH-Signal	MCH-Signal	Pin No.
1	GND	PWR_ON	170
2	PWR	Reserved for TDI	169
3	/PS1	Reserved for TDO	168
4	MP	Reserved for TRST#	167
5	GA0	Reserved for TMS	166
6	RESVD	Reserved for TCK	165
7	GND	GND	164
8	RESVD	TxFA-1+	163
9	PWR	TxFA-1-	162
10	GND	GND	161
11	TxFUA+	RxFA-1+	160
12	TxFUA-	RxFA-1-	159
13	GND	GND	158
14	RxFUA+	TxFA-2+	157
15	RxFUA-	TxFA-2-	156
16	GND	GND	155
17	GA1	RxFA-2+	154
18	PWR	RxFA-2-	153
19	GND	GND	152
20	TxFA-3+	TxFA-4+	151
21	TxFA-3-	TxFA-4-	150
22	GND	GND	149
23	RxFA-3+	RxFA-4+	148
24	RxFA-3-	RxFA-4-	147
25	GND	GND	146
26	GA2	TxFA-6+	145
27	PWR	TxFA-6-	144
28	GND	GND	143
29	TxFA-5+	RxFA-6+	142
30	TxFA-5-	RxFA-6-	141
31	GND	GND	140
32	RxFA-5+	TxFA-8+	139
33	RxFA-5-	TxFA-8-	138
34	GND	GND	137
35	TxFA-7+	RxFA-8+	136
36	TxFA-7-	RxFA-8-	135
37	GND	GND	134

Pin No.	MCH-Signal	MCH-Signal	Pin No.
38	RxFA-7+	/TMREQ	133
39	RxFA-7-	RSVD	132
40	GND	GND	131
41	/ENABLE	I2C_SCL	130
42	PWR	I2C_SDA	129
43	GND	GND	128
44	TxFA-9+	IPMB0-SCL-A	127
45	TxFA-9-	IPMB0-SDA-A	126
46	GND	GND	125
47	RxFA-9+	IPMB0-SCL-B	124
48	RxFA-9-	IPMB0-SDA-B	123
49	GND	GND	122
50	TxFA-10+	IPMBL-SCL-1	121
51	TxFA-10-	IPMBL-SDA-1	120
52	GND	GND	119
53	RxFA-10+	IPMBL-SCL-2	118
54	RxFA-10-	IPMBL-SDA-2	117
55	GND	GND	116
56	SCL_L	IPMBL-SCL-3	115
57	PWR	IPMBL-SDA-3	114
58	GND	GND	113
59	TxFA-11+	IPMBL-SCL-4	112
60	TxFA-11-	IPMBL-SDA-4	111
61	GND	GND	110
62	RxFA-11+	IPMBL-SCL-5	109
63	RxFA-11-	IPMBL-SDA-5	108
64	GND	GND	107
65	TxFA-12+	IPMBL-SCL-6	106
66	TxFA-12-	IPMBL-SDA-6	105
67	GND	GND	104
68	RxFA-12+	IPMBL-SCL-7	103
69	RxFA-12-	IPMBL-SDA-7	102
70	GND	GND	101
71	SDA_L	IPMBL-SCL-8	100
72	PWR	IPMBL-SDA-8	99
73	GND	GND	98
74	XOVER0+	IPMBL-SCL-9	97
75	XOVER0-	IPMBL-SDA-9	96
76	GND	GND	95
77	XOVER1+	IPMBL-SCL-10	94
78	XOVER1-	IPMBL-SDA-10	93

Pin No.	MCH-Signal	MCH-Signal	Pin No.
79	GND	GND	92
80	XOVER2+	IPMBL-SCL-11	91
81	XOVER2-	IPMBL-SDA-11	90
82	GND	GND	89
83	/PS0	IPMBL-SCL-12	88
84	PWR	IPMBL-SDA-12	87
85	GND	GND	86

Table 13-5 MCH Connector Tongue 1

Appendix A 6 NAT-MCH-Clock Connector Tongue 2

Pin No.	MCH-Signal	MCH-Signal	Pin No.
1	GND	GND	170
2	RSVD	RSVD	169
3	RSVD	RSVD	168
4	GND	GND	167
5	RSVD	RSVD	166
6	RSVD	RSVD	165
7	GND	GND	164
8	CLK3_Tx+	CLK3_Rx+	163
9	CLK3_Tx-	CLK3_Rx-	162
10	GND	GND	161
11	CLK1_Tx+	CLK1_Rx+	160
12	CLK1_Tx-	CLK1_Rx-	159
13	GND	GND	158
14	TxFB-1+	RxFB-1+	157
15	TxFB-1-	RxFB-1-	156
16	GND	GND	155
17	TxFB-2+	RxFB-2+	154
18	TxFB-2-	RxFB-2-	153
19	GND	GND	152
20	TxFB-3+	RxFB-3+	151
21	TxFB-3-	RxFB-3-	150
22	GND	GND	149
23	TxFB-4+	RxFB-4+	148
24	TxFB-4-	RxFB-4-	147
25	GND	GND	146
26	TxFB-5+	RxFB-5+	145
27	TxFB-5-	RxFB-5-	144
28	GND	GND	143
29	TxFB-6+	RxFB-6+	142
30	TxFB-6-	RxFB-6-	141
31	GND	GND	140
32	CLK3-1+	CLK3-7+	139
33	CLK3-1-	CLK3-7-	138
34	GND	GND	137
35	CLK3-2+	CLK3-8+	136
36	CLK3-2-	CLK3-8-	135
37	GND	GND	134

Pin No.	MCH-Signal	MCH-Signal	Pin No.
38	CLK3-3+	CLK3-9+	133
39	CLK3-3-	CLK3-9-	132
40	GND	GND	131
41	CLK3-4+	CLK3-10+	130
42	CLK3-4-	CLK3-10-	129
43	GND	GND	128
44	CLK3-5+	CLK3-11+	127
45	CLK3-5-	CLK3-11-	126
46	GND	GND	125
47	CLK3-6+	CLK3-12+	124
48	CLK3-6-	CLK3-12-	123
49	GND	GND	122
50	CLK1-1+	CLK2-1+	121
51	CLK1-1-	CLK2-1-	120
52	GND	GND	119
53	CLK1-2+	CLK2-2+	118
54	CLK1-2-	CLK2-2-	117
55	GND	GND	116
56	CLK1-3+	CLK2-3+	115
57	CLK1-3-	CLK2-3-	114
58	GND	GND	113
59	CLK1-4+	CLK2-4+	112
60	CLK1-4-	CLK2-4-	111
61	GND	GND	110
62	CLK1-5+	CLK2-5+	109
63	CLK1-5-	CLK2-5-	108
64	GND	GND	107
65	CLK1-6+	CLK2-6+	106
66	CLK1-6-	CLK2-6-	105
67	GND	GND	104
68	CLK1-7+	CLK2-7+	103
69	CLK1-7-	CLK2-7-	102
70	GND	GND	101
71	CLK1-8+	CLK2-8+	100
72	CLK1-8-	CLK2-8-	99
73	GND	GND	98
74	CLK1-9+	CLK2-9+	97
75	CLK1-9-	CLK2-9-	96
76	GND	GND	95
77	CLK1-10+	CLK2-10+	94
78	CLK1-10-	CLK2-10-	93

Pin No.	MCH-Signal	MCH-Signal	Pin No.
79	GND	GND	92
80	CLK1-11+	CLK2-11+	91
81	CLK1-11-	CLK2-11-	90
82	GND	GND	89
83	CLK1-12+	CLK2-12+	88
84	CLK1-12-	CLK2-12-	87
85	GND	GND	86

Table 13-6 MCH Connector Tongue 2

Appendix A 7 NAT-MCH Hub-Module Connector Tongue 3

Pin No.	MCH-Signal	MCH-Signal	Pin No.
1	GND	GND	170
2	RSVD	RSVD	169
3	RSVD	RSVD	168
4	GND	GND	167
5	RSVD	RSVD	166
6	RSVD	RSVD	165
7	GND	GND	164
8	TxFUD+	RxFUD+	163
9	TxFUD-	RxFUD-	162
10	GND	GND	161
11	TxFUE+	RxFUE+	160
12	TxFUE-	RxFUE-	159
13	GND	GND	158
14	TxFD1+	RxFD1+	157
15	TxFD1-	RxFD1-	156
16	GND	GND	155
17	TxFE1+	RxFE1+	154
18	TxFE1-	RxFE1-	153
19	GND	GND	152
20	TxFF1+	RxFF1+	151
21	TxFF1-	RxFF1-	150
22	GND	GND	149
23	TxFG1+	RxFG1+	148
24	TxFG1-	RxFG1-	147
25	GND	GND	146
26	TxFD2+	RxFD2+	145
27	TxFD2-	RxFD2-	144
28	GND	GND	143
29	TxFE2+	RxFE2+	142
30	TxFE2-	RxFE2-	141
31	GND	GND	140
32	TxFF2+	RxFF2+	139
33	TxFF2-	RxFF2-	138
34	GND	GND	137
35	TxFG2+	RxFG2+	136
36	TxFG2-	RxFG2-	135
37	GND	GND	134

Pin No.	MCH-Signal	MCH-Signal	Pin No.
38	TxFD3+	RxFD3+	133
39	TxFD3-	RxFD3-	132
40	GND	GND	131
41	TxFE3+	RxFE3+	130
42	TxFE3-	RxFE3-	129
43	GND	GND	128
44	TxFF3+	RxFF3+	127
45	TxFF3-	RxFF3-	126
46	GND	GND	125
47	TxFG3+	RxFG3+	124
48	TxFG3+	RxFG3-	123
49	GND	GND	122
50	TxFD4+	RxFD4+	121
51	TxFD4-	RxFD4-	120
52	GND	GND	119
53	TxFE4+	RxFE4+	118
54	TxFE4-	RxFE4-	117
55	GND	GND	116
56	TxFF4+	RxFF4+	115
57	TxFF4-	RxFF4-	114
58	GND	GND	113
59	TxFG4+	RxFG4+	112
60	TxFG4-	RxFG4-	111
61	GND	GND	110
62	TxFD5+	RxFD5+	109
63	TxFD5-	RxFD5-	108
64	GND	GND	107
65	TxFE5+	RxFE5+	106
66	TxFE5-	RxFE5-	105
67	GND	GND	104
68	TxFF5+	RxFF5+	103
69	TxFF5-	RxFF5-	102
70	GND	GND	101
71	TxFG5+	RxFG5+	100
72	TxFG5-	RxFG5-	99
73	GND	GND	98
74	TxFD6+	RxFD6+	97
75	TxFD6-	RxFD6-	96
76	GND	GND	95
77	TxFE6+	RxFE6+	94
78	TxFE6-	RxFE6-	93

Pin No.	MCH-Signal	MCH-Signal	Pin No.
79	GND	GND	92
80	TxFF6+	RxFF6+	91
81	TxFF6+	RxFF6-	90
82	GND	GND	89
83	TxFG6+	RxFG6+	88
84	TxFG6-	RxFG6-	87
85	GND	GND	86

Table 13-7 MCH Connector Tongue 3

Appendix A 8 NAT-MCH Hub-Module Connector Tongue 4

Pin No.	MCH-Signal	MCH-Signal	Pin No.
1	GND	GND	170
2	RSVD	RSVD	169
3	RSVD	RSVD	168
4	GND	GND	167
5	RSVD	RSVD	166
6	RSVD	RSVD	165
7	GND	GND	164
8	TxFUF+	RxFUF+	163
9	TxFUF-	RxFUF-	162
10	GND	GND	161
11	TxFUG+	RxFUG+	160
12	TxFUG-	RxFUG-	159
13	GND	GND	158
14	TxFD7+	RxFD7+	157
15	TxFD7-	RxFD7-	156
16	GND	GND	155
17	TxFE7+	RxFE7+	154
18	TxFE7-	RxFE7-	153
19	GND	GND	152
20	TxFF7+	RxFF7+	151
21	TxFF7-	RxFF7-	150
22	GND	GND	149
23	TxFG7+	RxFG7+	148
24	TxFG7-	RxFG7-	147
25	GND	GND	146
26	TxFD8+	RxFD8+	145
27	TxFD8-	RxFD8-	144
28	GND	GND	143
29	TxFE8+	RxFE8+	142
30	TxFE8-	RxFE8-	141
31	GND	GND	140
32	TxFF8+	RxFF8+	139
33	TxFF8-	RxFF8-	138
34	GND	GND	137
35	TxFG8+	RxFG8+	136
36	TxFG8-	RxFG8-	135
37	GND	GND	134

Pin No.	MCH-Signal	MCH-Signal	Pin No.
38	TxFD9+	RxFD9+	133
39	TxFD9-	RxFD9-	132
40	GND	GND	131
41	TxFE9+	RxFE9+	130
42	TxFE9-	RxFE9-	129
43	GND	GND	128
44	TxFF9+	RxFF9+	127
45	TxFF9-	RxFF9-	126
46	GND	GND	125
47	TxFG9+	RxFG9+	124
48	TxFG9+	RxFG9-	123
49	GND	GND	122
50	TxFD10+	RxFD10+	121
51	TxFD10-	RxFD10-	120
52	GND	GND	119
53	TxFE10+	RxFE10+	118
54	TxFE10-	RxFE10-	117
55	GND	GND	116
56	TxFF10+	RxFF10+	115
57	TxFF10-	RxFF10-	114
58	GND	GND	113
59	TxFG10+	RxFG10+	112
60	TxFG10-	RxFG10-	111
61	GND	GND	110
62	TxFD11+	RxFD11+	109
63	TxFD11-	RxFD11-	108
64	GND	GND	107
65	TxFE11+	RxFE11+	106
66	TxFE11-	RxFE11-	105
67	GND	GND	104
68	TxFF11+	RxFF11+	103
69	TxFF11-	RxFF11-	102
70	GND	GND	101
71	TxFG11+	RxFG11+	100
72	TxFG11-	RxFG11-	99
73	GND	GND	98
74	TxFD12+	RxFD12+	97
75	TxFD12-	RxFD12-	96
76	GND	GND	95
77	TxFE12+	RxFE12+	94
78	TxFE12-	RxFE12-	93

Pin No.	MCH-Signal	MCH-Signal	Pin No.
79	GND	GND	92
80	TxFF12+	RxFF12+	91
81	TxFF12+	RxFF12-	90
82	GND	GND	89
83	TxFG12+	RxFG12+	88
84	TxFG12-	RxFG12-	87
85	GND	GND	86

Table 13-8 MCH Connector Tongue 4

Appendix B MCH access from a different subnetwork

This section describes how to setup the MCH in a subnetwork and to access the MCH from a different subnetwork. Please note that this appendix describes the network configuration using the defaults stored in the so-called 'board configuration' EEPROM. Therefore the following explanations only apply if the IP address source in the MCH configuration is set to 'board configuration'.

Appendix B 1 Network setup:

The following scheme shows the example network configuration that is used in this description:

```
MCH IP Address: 192.168.0.134
                ^
                |
                | network 192.168.0.x (first subnetwork)
                |
                v
Linux Gateway IP Adress: 192.168.0.70   first port (eth1)
Linux Gateway IP Adress: 132.147.160.70 second port (eth0)
                ^
                |
                | network 132.147.160.x (second subnetwork)
                |
                v
Control host IP address: 132.147.160.47
        default gateway: 132.147.160.70
```

The term 'gateway' is used in that manner that a gateway changes IP addresses (using Network Address Translation, see below) whereas a 'router' does not change IP addresses.

Please note that the gateway must be reachable from both the MCH and the control host on different subnetworks that are differentiated by the network masks.

Appendix B 2 MCH IP address configuration:

To use the MCH in a subnetwork only one IP address needs to be configured. However, the second IP address might be used for a different purpose. The MCH IP address can be configured using the CLI command 'ip' and the configuration can be printed using the command 'bi':

```
nat> bi

Board Configuration Information:
```

```
Board Identifier: 0b0321-0285
Serial Number   : 285
Manufacturer ID : 01
Board Code      : 0b03
Layout Version : 2.1
Revison Code    : 071025
CPU             : Coldfire / 162 MHz
DRAM            : 32 MB
SRAM            : 32 KB
EEPROM          : 32 MB
IEEE Address    : 00-40-42-0b-01-1d
```

```
TCP/IP Configuration:
   IP Address   : 192.168.0.134
   IP Net Mask  : 255.255.255.0
   IP Broadcast : 192.168.0.255
   IP Gateway   : 192.168.0.70
   2. IP Address   : 0.0.0.0
➢    IP Net Mask  : 0.0.0.0
➢    IP Broadcast : 0.0.0.0
   2. IP Gateway   : 0.0.0.0
Boot String     : 0xd0100008
```

Issuing the 'route' CLI command on the MCH gives:

```
nat> route
--- IP routing tables ---
Destination       Gateway           Flags     Ref     Use  Interface
0.0.0.0           192.168.0.70      UG          1       3  nat0
192.168.0.0       *                 U           0       0  nat0
--- 2 routings ---
```

Appendix B 3 Linux Gateway configuration:

The linux gateway was configured to forward IP packets and to use NAT (Network Address Translation, also known as 'masquerading'). The linux system (kernel 2.4 and 2.6) can be configured as gateway by the commands:

```
root# echo 1 > /proc/sys/net/ipv4/ip_forward
root# iptables –A POSTROUTING –t nat –o eth1 –j MASQUERADE
```

To disable the gateway configuration enter:

```
root# iptables –D POSTROUTING –t nat –o eth1 –j MASQUERADE
root# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Appendix B 4 Test1: ping between MCH and gateway

This test shows whether the MCH can communicate with the gateway and vice versa. An echo request is issued to the remote system (using the MCH CLI command 'ping') which should send back an echo reply message. Only if packets are transmitted and received correctly from both systems the communication between the systems work.

Ping the gateway from the MCH:

```
nat> ping 192.168.0.70
192.168.0.134->192.168.0.70 Echo Request (56 bytes) seq 0
192.168.0.70->192.168.0.134 Echo Reply   (56 bytes) seq 0 ttl 64
192.168.0.134->192.168.0.70 Echo Request (56 bytes) seq 1
192.168.0.70->192.168.0.134 Echo Reply   (56 bytes) seq 1 ttl 64
192.168.0.134->192.168.0.70 Echo Request (56 bytes) seq 2
192.168.0.70->192.168.0.134 Echo Reply   (56 bytes) seq 2 ttl 64
--- 192.168.0.70 ping statistics ---
  packets: tx 3   rx 3
```

ping the MCH from the gateway:

```
root# ping 192.168.0.134
PING 192.168.0.134 (192.168.0.134) 56(84) bytes of data.
64 bytes from 192.168.0.134: icmp_seq=1 ttl=255 time=0.276 ms
64 bytes from 192.168.0.134: icmp_seq=2 ttl=255 time=0.247 ms
64 bytes from 192.168.0.134: icmp_seq=3 ttl=255 time=0.249 ms
64 bytes from 192.168.0.134: icmp_seq=4 ttl=255 time=0.235 ms
--- 192.168.0.134 ping statistics ---
   3  packets transmitted, 4 received, 0% packet loss, time 3001ms
rtt min/avg/max/mdev = 0.235/0.244/0.258/0.008 ms
```

Appendix B 5 Test2: ping between control host and gateway

This test shows whether the control host can communicate with the gateway and vice versa using the 'ping' tool.

Ping the gateway from the control host:

```
control:~$ ping 132.147.160.70
PING 132.147.160.70 (132.147.160.70) 56(84) bytes of data.
64 bytes from 132.147.160.70: icmp_seq=1 ttl=64 time=0.237 ms
64 bytes from 132.147.160.70: icmp_seq=2 ttl=64 time=0.193 ms
64 bytes from 132.147.160.70: icmp_seq=3 ttl=64 time=0.192 ms
--- 132.147.160.70 ping statistics ---
➢    packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.192/0.207/0.237/0.024 ms
```

ping the control host from the gateway:

```
root@natcg2# ping 132.147.160.47
PING 132.147.160.47 (132.147.160.47) 56(84) bytes of data.
64 bytes from 132.147.160.47: icmp_seq=1 ttl=64 time=0.232 ms
64 bytes from 132.147.160.47: icmp_seq=2 ttl=64 time=0.193 ms
64 bytes from 132.147.160.47: icmp_seq=3 ttl=64 time=0.195 ms

--- 132.147.160.47 ping statistics ---
➢    packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.193/0.206/0.232/0.024 ms
```

Appendix B 6 Test3: ping between control host and MCH:

This test shows whether the control host can communicate with the MCH and vice versa using the 'ping' tool.

Ping the MCH from the control host:

```
control:~$ ping 192.168.0.134
PING 192.168.0.134 (192.168.0.134) 56(84) bytes of data.
64 bytes from 192.168.0.134: icmp_seq=1 ttl=254 time=4.73 ms
64 bytes from 192.168.0.134: icmp_seq=2 ttl=254 time=0.382 ms
64 bytes from 192.168.0.134: icmp_seq=3 ttl=254 time=0.342 ms
64 bytes from 192.168.0.134: icmp_seq=4 ttl=254 time=0.346 ms
--- 192.168.0.134 ping statistics ---
   4  packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.342/1.450/4.732/1.895 ms
```

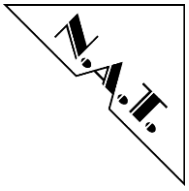
and a network sniffer dump gives:

```
No. Time      Source          Destination     Prot  Info
25  5.616704  132.147.160.47  192.168.0.134   ICMP  Echo (ping) request
26  5.616755  192.168.0.70    192.168.0.134   ICMP  Echo (ping) request
27  5.616939  192.168.0.134   192.168.0.70    ICMP  Echo (ping) reply
28  5.616953  192.168.0.134   132.147.160.47  ICMP  Echo (ping) reply
```

Note that a ping request issued from the control host is received by the gateway and then delivered to MCH, the ping response is first sent to the gateway and then delivered to the control host. In both cases network address translation is performed by the gateway.

Ping the the control host from the MCH:

```
nat> ping 132.147.160.47
192.168.0.134->132.147.160.47 Echo Request (56 bytes) seq 0
132.147.160.47->192.168.0.134 Echo Reply   (56 bytes) seq 0 ttl 63
192.168.0.134->132.147.160.47 Echo Request (56 bytes) seq 1
132.147.160.47->192.168.0.134 Echo Reply   (56 bytes) seq 1 ttl 63
192.168.0.134->132.147.160.47 Echo Request (56 bytes) seq 2
132.147.160.47->192.168.0.134 Echo Reply   (56 bytes) seq 2 ttl 63
```



```
--- 132.147.160.47 ping statistics ---
  packets: tx 3   rx 3
```

and the network sniffer dump gives:

```
   No. Time      Source          Destination     Prot  Info
   46  12.307430 192.168.0.134   132.147.160.47  ICMP  Echo (ping) request
➢     12.307485 132.147.160.70  132.147.160.47  ICMP  Echo (ping) request
➢     12.307643 132.147.160.47  132.147.160.70  ICMP  Echo (ping) reply
   49  12.307658 132.147.160.47  192.168.0.134   ICMP  Echo (ping) reply
```

If the ping test between the control host and the MCH (or vice versa) fails but Test1 and Test 2 succeeded there is probably something wrong with the gateway configuration.

Appendix C N.A.T. defined OEM IPMI Messages

Appendix C 1 IPMI messages to read and write register

The following chapter lists the IPMI extensions defined by N.A.T. for reading and writing registers on the hub/clock mezzanine modules:

Read Register Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Register Number
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00
	3	Register Read value

Write Register Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Register Number
	4	Register Write value
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

Set Bit Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Register Number
	4	Bit Number (0-7) to set
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

Clear Bit Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Register Number
	4	Bit Number (0-7) to clear

Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

Pulse Bit low Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Register Number
	4	Bit Number (0-7) to pulse low
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

Pulse Bit high Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Register Number
	4	Bit Number (0-7) to pulse high
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

Read Register Long Word Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Device Number
	4	Register Number [3]
	5	Register Number [2]
	6	Register Number [1]
	7	Register Number [0]
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00
	3	Read Register Value [0]
	4	Read Register Value [1]
	5	Read Register Value [2]
	6	Read Register Value [3]

Write Register Long Word Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Device Number
	4	Register Number [3]
	5	Register Number [2]
	6	Register Number [1]
	7	Register Number [0]

	8	Data[0]
	9	Data[1]
	10	Data[2]
	11	Data[3]
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

Read Register Long Word Indirect Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Device Number
	4	Register Number [0]
	5	Register Number [1]
	6	Register Number [2]
	7	Register Number [3]
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00
	3	Read Register Value [0]
	4	Read Register Value [1]
	5	Read Register Value [2]
	6	Read Register Value [3]

Write Register Long Word Indirect Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Device Number
	4	Register Number [0]
	5	Register Number [1]
	6	Register Number [2]
	7	Register Number [3]
	8	Data[0]
	9	Data[1]
	10	Data[2]
	11	Data[3]
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

Read Block Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Device Number
	4	Length
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

3-N	Data

Write Block Command:

	Byte	Data Field
Request Data	1	PICMG Identifier = 0x00
	2	0x00
	3	Device Number
	4	Length
	5-N	Data
Response Data	1	Completion Code
	2	PICMG Identifier = 0x00

Controller/Vendor specific extensions – NetFn is IPMI_NETFN_CONTR0 (IPMI)

#define IPMI_CMD_READ_REG	0x01
#define IPMI_CMD_WRITE_REG	0x02
#define IPMI_CMD_SET_BIT	0x03
#define IPMI_CMD_CLR_BIT	0x04
#define IPMI_CMD_PULSE_BITL	0x05
#define IPMI_CMD_PULSE_BITH	0x06
#define IPMI_CMD_READ_REG_LW	0x07
#define IPMI_CMD_WRITE_REG_LW	0x08
#define IPMI_CMD_READ_REG_LWI	0x09
#define IPMI_CMD_WRITE_REG_LWI	0x0a
#define IPMI_CMD_READ_BLCK	0x0b
#define IPMI_CMD_WRITE_BLCK	0x0c

The I2C addresses of the Hub/clock IPMI devices are:

Device	MCH Site 1	MCH Site 2
Clock Module	0x14	0x18
Hub Module	0x16	0x1a

Appendix C 2 Register access using ipmitool

This chapter gives a list of example calls for ipmitool to access registers on the clock or hub modules. Please note that register access IPMI messages have to be 'double bridged' to be delivered to the related modules which requires at least ipmitool version 1.8.11.
Read the Clock Module board identifier:

- `ipmitool -H 132.147.160.251 -P "" -t 0x82 -T 0x14 raw 0x30 0x01 0x00 0x00 0x00`

Read the Clock Module PCB version:

- `ipmitool -H 132.147.160.251 -P "" -t 0x82 -T 0x14 raw 0x30 0x01 0x00 0x00 0x01`

Read the Clock Module reference 0 selection:

➢ `ipmitool –H 132.147.160.251 –P "" –t 0x82 –T 0x14 raw 0x30 0x01 0x00 0x00 0x04`

Write the Clock Module reference 0 selection to 0xaa:

➢ `ipmitool –H 132.147.160.251 –P « » –t 0x82 –t 0x14 raw 0x30 0x02 0x00 0x00 0x04 0xaa`

Appendix C 3

It is also possible to access any AMC, power module, cooling unit etc. by ipmitool via the MCH. In this example, the firmware of power module 1 shall be updated with ipmitool. Please refer to the ipmitool documentation for further parameters.
« ipmitool –I lan –H 192.168.1.146 –A none –T 0x82 –B 0 –t 0xc2 –b 7 hpm upgrade fw.hpm »

Parameters used :

-I lan	//interface : lan
-H 192.168.1.146	// ip-address : 192.168.1.146
-A none	// authentication : none
-T 0x82	// target address of the carrier (80h + (Carrier Number * 2))
-B 0	// bus id of the carrier (IPMB-0)
-t 0xc2	// target address (power module 1 has IPMB-L address 0xc2)
-b 7	// bus is (IPMB-L)

hpm upgrade firmware.hpm

Other possible commands are :

targetcap	// shows target capabilities
hpm compprop <id> <select>	// shows component properties
<select>=1	show firmware revision
<select>=2	show component type
hpm upgstatus	// status of the last long duration command
hpm activate	// activate upgraded firmware

The I2C addresses of several FRU devices are:

Device	target address
AMC 1	0x72
AMC 2	0x74
…	…
AMC 12	0x88
CU 1	0xa8
CU 2	0xaa
PM 1	0xc2
PM 2	0xc4
PM 3	0xc6
PM 4	0xc8

Appendix D Adapting a Host App from Compatible to Native mode

Appendix D 1 Preface

The NAT-MCH firmware has changed over time. Starting with firmware release 2.5 a new concept called ShM/CM-Separation was introduced. To keep all older software operational with this new firmware the configuration option "Compatible Mode" was added. Enabling this compatible mode forced the MCH firmware to behave in exactly the same way as it did with firmware releases before 2.5: there was only one single ShM/CM unit for the host software.
This document shall describe the necessary changes that need to be made to make a MicroTCA host application compatible to the NAT-MCH native mode.

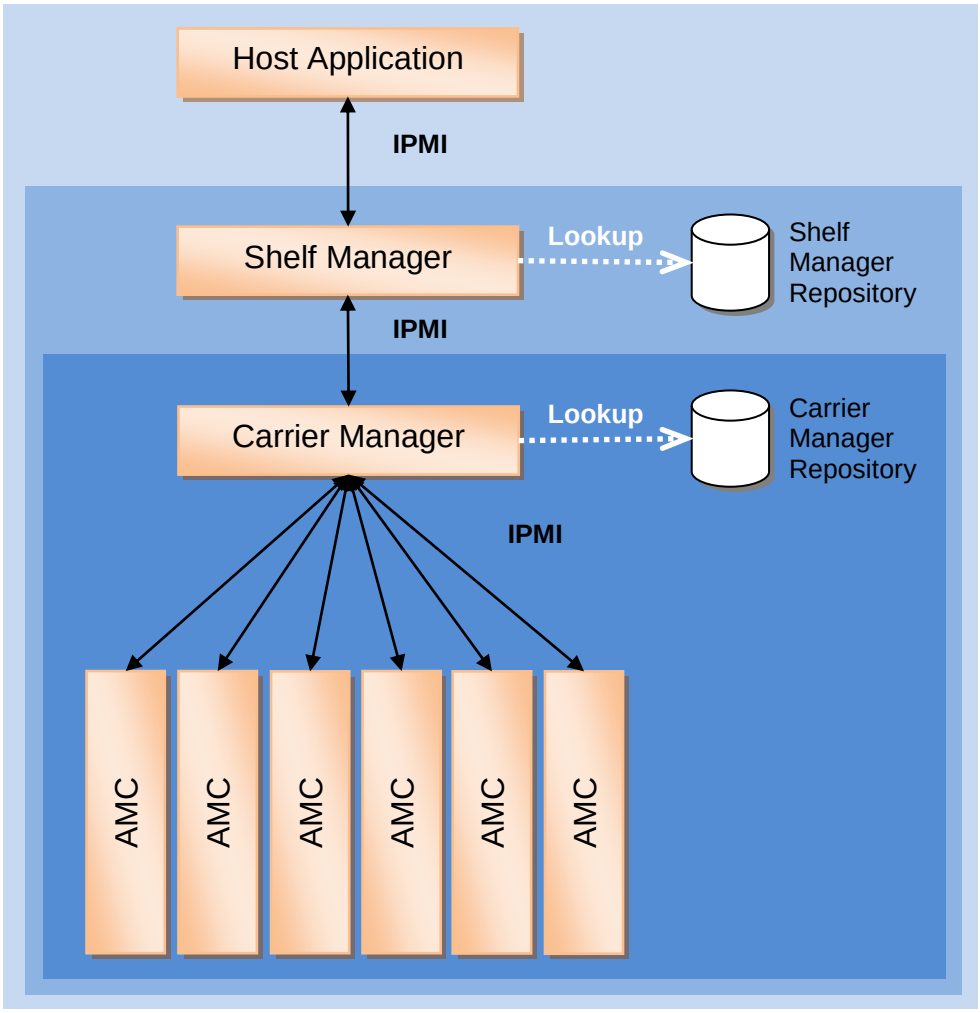
Appendix D 2 Changes of the MCH Firmware

13.1 Overview

The main differences between the compatible and the native mode are as follows:

Compatible Mode	*Native Mode*
Single Shelf/Carrier Manager	Separate Shelf and Carrier Manager
AMC access via single bridged messages	AMC access via double bridged messages
Resource Browsing using Shelf/Carrier Manager repository	Resource browsing using a combination of the Shelf and the Carrier Manager repository
SEL uses AMC sensor numbering	SEL uses Shelf Manager sensor numbering
Hot swap detection via SEL	Hot swap detection via SEL and Carrier Manager repository

All these new features of the native mode will be covered more detailed in the following sections of this document. The overall structure of the NAT-MCH firmware from release 2.5 on is illustrated in the following diagram:



To access the different levels of a system, a host application needs to know where to send its IPMI messages to:

- It can talk to the Shelf Manager directly using standard IPMI messages.
- To talk to the Carrier Manager the host application needs to send singly encapsulated IPMI messages to the Shelf Manager. To investigate the Carrier Manager Repository it needs to set up a proper SDR message from the Shelf to the Carrier Manager (e.g. Get SDR). This request is then encapsulated into a Send Message Request message that is being addressed from the host application to the Shelf Manager.
- If the host application needs to investigate the AMC module directly it needs to perform another layer of encapsulation.

Also have a look at section 13.3 about double bridged messages.

13.2 Separate Shelf and Carrier Manager

From MCH firmware release 2.5 on a separate, external Shelf Manager (ShM) is supported. Therefore a logical separation of the Shelf Manager and the Carrier Manager (CM)

functionality was necessary. Both entities have their own device repositories which are maintained independently from each other. The Carrier Manager creates and updates its repository by scanning its hardware devices. The Shelf Manager periodically scans the Carrier Manager's repositories to update all changes.
As the CM repository adds a devices as soon as it is visible to the management interface (hotswap state M1 and above), the ShM repository will add new devices only if they reside in hotswap state M2 or above. A consequence of this is that a newly inserted AMC module with a pulled-out hotswap handle will be visible in the CM repository but not in the one of the ShM. This has also consequences when the module is being pulled out of the chassis: the ShM repository removes the device after the M6 to M1 transition although the hardware is still in the system. The only way to get the whole picture from a host application is to periodically scan the CM repository.

13.3 AMC access via double bridged messages

According to the IPMI specification every media change of an IPMI message needs to be encapsulated into a SendMessage container.
When operating in compatible mode this means that a host application had to use single bridged messages to communicate with all devices on the IPMB-L (AMCs, MCHs) and IPMB-0 (Cus, PMs). The SendMessage request with its encapsulated message was directed to address 0x20, unpacked by the ShM/CM and then sent to the inquired device. As the device response arrived the ShM/CM encapsulated it into a SendMessage response which is then sent back to the host application.
With introduction of the native mode and its new logical level, the IPMI message handling changed as well. The concept is explained below by an example: sending GetDeviceId request to AMC1 (FRU 5).

1. The host application constructs an IPMI request message for AMC1, IPMB-L address 0x72.
2. This request is encapsulated in the first SendMessage request directed to the (first) Carrier Manager. Its address on the ShM-CM-interface is calculated with the formula *0x80 + (Carrier Number * 2)*. (See [1], clause 3.4.3 "Shelf-Carrier Manager Interface" for details.)
3. This request is encapsulated once more in another SendMessage request directed to the Shelf Manager. Its address is 0x20.
4. This whole message is then send to the Shelf Manager. It removes the outer SendMessage request message and processes the result by sending it to the (first) Carrier Manager.
5. The Carrier Manager removes the leftover SendMessage and processes the result by sending the GetDeviceId request to the appropriate AMC module.
6. After the Carrier manager has received the GetDeviceId response from the AMC module it encapsulates it into a SendMessage response for the Shelf Manager.
7. The Shelf Manager encapsulates the response into another SendMessage response message for the host application.
8. The Shelf Manager sends the complete response to the host application.

13.4 Resource browsing using a combination of Shelf Manager and Carrier Manager Repository

The ShM repository shows all devices that have a hotswap state equal or above M2. This means that an AMC module with pulled hotswap handle will not be visible in the ShM repository. The only way to find all devices that are currently connected to the system is to scan both the ShM and the CM repositories. A common algorithm for a host application would look like this:

- On startup of the host application the ShM repository is scanned. The application shall scan all records for FruDeviceLocator records.
- For every FruDeviceLocator record a new FRU device must be generated.
- To find the corresponding sensor entries of this FRU device save the EntityID and EntityInstance of the FruDeviceLocator record. Scan the ShM repository for records with this matching pair.
- Periodically read the SEL and process its events. These events correspond to the ShM repository entries.
- Periodically scan the CM repository to find all devices in hotswap state M1. Synchronize both repositories entries using the FRU ID of an entry.

Note: This complicated procedure is necessary to make the behavior of the Shelf Manager exactly how it is required by the specifications.

13.5 SEL uses Shelf Manager Sensor Numbering

There is no sensor numbering that is valid in every area of a MicroTCA system. Every administrative layer has its own numbering.

- The AMC modules perform a sensor numbering on board level. Although it is not requested all board sensor numbers usually start with 1.
- The Carrier Manager needs to merge all sensors of all carrier resources into one single repository. It is therefore necessary to perform a mapping between the board-level sensor numbering to a carrier-wide sensor numbering. The 8-bit limitation of the sensor ID value in the SDRs is not sufficient (allowing only 256 sensor for a carrier at max); to extend this sensor ID range, three of the four possible LUN numbers are used for sensor addressing (LUN 2 is reserved for message bridging).
- The Shelf Manager performs another layer of sensor ID mapping the same way as the Carrier Manager does.

As a host application will usually read out the SEL from the ShM all its events will use data from the ShM repository. Although it is still possible to read out sensor data directly it is not recommended as it is impossible for a host application to retrieve the board-local sensor ID from the ShM sensor ID. According to the specifications there is no way for a host application to map a ShM repository sensor to the corresponding sensor of the CM or AMC board repository. All sensors related requests need to be directed to the ShM that will retrieve the appropriate data.

13.6 Hotswap Detection via SEL and Carrier Manager Repository

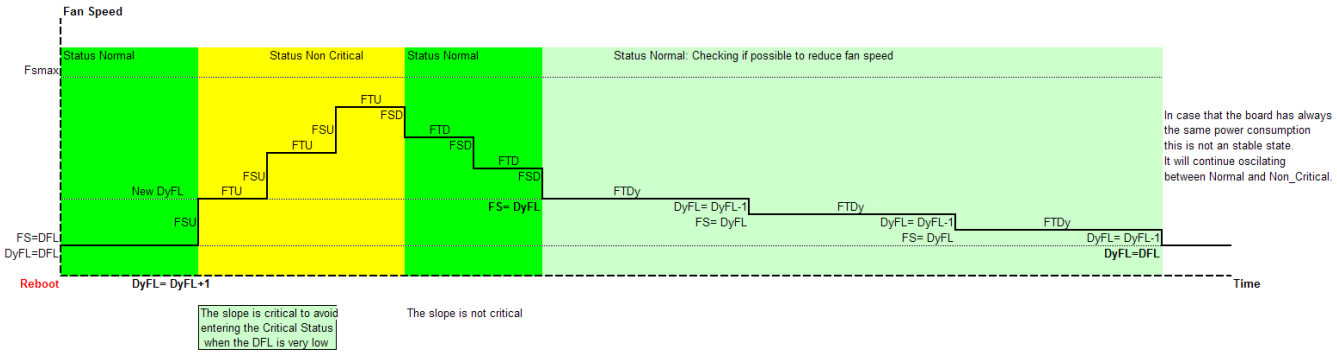
As mentioned above the ShM and CM repository handle detect resources in a different manner. The CM adds all detected resources to its repository once it is detected which means that all devices show up in the CM repository if they are in hotswap state M1 or higher. The ShM repository only contains devices that are in hotswap state M2 or higher.
This implies tiny differences on when a device will show up in which repository. During a typical hotswap cycle the following information is contained in the ShM and the CM repositories:

Action	*Transition*	*Visible in CM repository?*	*Visible in ShM repository?*
Board not yet inserted.	–	No	No
Board insertion, hotswap handle is open ("pulled out").	M0->M1	Yes	No
Close hotswap handle ("push in").	M1->M2->M3->M4	Yes	Yes
Open hotswap handle ("pull out").	M4->M5->M6->M1	Yes	No
Board is removed from chassis.	M1->M0	No	No

The host application can use the ShM SEL to monitor all state transitions from M2 and above. To monitor the state transitions from M0 to M1 (inclusive), it is necessary for the host application to periodically scan the Carrier Manager repository.

Appendix E Alternative cooling unit scheme

The standard cooling unit scheme increases the fan speed of the cooling units to 100% as soon as temperature event is sent by one of the resources in the system. Sometime this could be annoying especially when testing a system in the Lab as some systems have quite noisy fans. The alternative cooling unit scheme controls the fans as shown in the picture below to find the best fan speed for the current system configuration.



Glossary:

Fan Speed (FS): Actual Fan Speed
Maximum Fan Speed (FSmax): Fans at 100% RPMs.
Default_Fan_Level (DFL): The Fan Speed after a MCH reboot (configurable)
Dynamic_Fan_Level (DyFL): The new Fan Speed after a Temp. Sensor alarm.
Fan_Step_Up (FSU)
Fan_Time_Up (FTU)
Fan_Step_Down (FSD)
Fan_Time_Down (FTD)
Fan_Time_Down_Dy (FTDy)

Possible MCH Status:

NORMAL: No Temp. Alarm.
Non_Critical: At least one Temp. Sensor has reported that the upper noncritical threshold has been crossed.
Critical: At least one Temp. Sensor has reported that the upper critical threshold has been crossed.
Non_Recoverable: At least one Temp. Sensor has reported that the non-recoverable threshold has been crossed.

Fan control description:

After an MCH reboot:
MCH Status= Normal
Fan Speed (FS) = DFL

DyFL = DFL

Non_Critical Event:

DyFL= DyFL+1
[FS=FS + FSU] each FTU seconds until FS=FSmax

Normal:

[FS=FS - FSD] each FTD seconds until FS= DyFL

When [FS= DyFL] then

Critical Event:

FS= Fsmax until normal status
DyFL= DyFL+1

Appendix F Updating the Backplane EEPROM via MCH CLI

A combined backplane FRU image file can be downloaded via the CLI diagnose menu of the MCH. All chassis individual information (e.g. like the serial number,…) have usually to be picked up before downloading the new combined backplane image file. Typically this information has to be entered manually afterwards again (e.g. with NATView). The EEPROM of the backplane can be updated with a combined backplane FRU image file as follow:

1. Set up a CLI connection to the MCH (e.g. via the RS-232 interface (baud rate 19200, data: 8bit, parity: none, stop: 1bit), USB or telnet)
2. Type diag when the connection to the MCH is set up
3. Choose submenu "[2] : (submenu) UPDATE menu" in diag menu
4. Select "[3] : update backplane EEPROM"
5. Enter the complete path and file name of the EEPROM file (located on a TFTP server), e.g. 192.168.137.70:/home/download/released/combined_backplane_fru.bin
6. Confirm "Enter EEPROM I2C address (RET=164/0xa4):" just with ENTER (I2C address is 0xa4))

The following response is shown in the CLI when the file is written successfully to the EEPROM:

writing EEPROM contents - please wait ...
EEPROM contents successfully updated
UPDATE (RET=0/0x0):

In case of a redundant system the backplane has a separate EEPROM for each MCH. Therefore the update procedure has to be done on both MCHs separately.

Appendix G PCIe Hot Plug Support (optional)

In case that the NAT-MCH is equipped with an optional PCIe Hub module the PCIe hot plug support can be activated via PCIe Switch Configuration parameter "hot plug support". Figure 17 illustrates how the MCH firmware (>= V2.16; HUB-PCIe-x24/48 AVR firmware >= V1.9 and FPGA image >= V1.5; HUB-PCIe-x80) is serving the following hot plug signals with respect to the AMC M states:

- HPPresent
- HPMrl
- HPButton
- HPPwrGood
- HPPwrEn

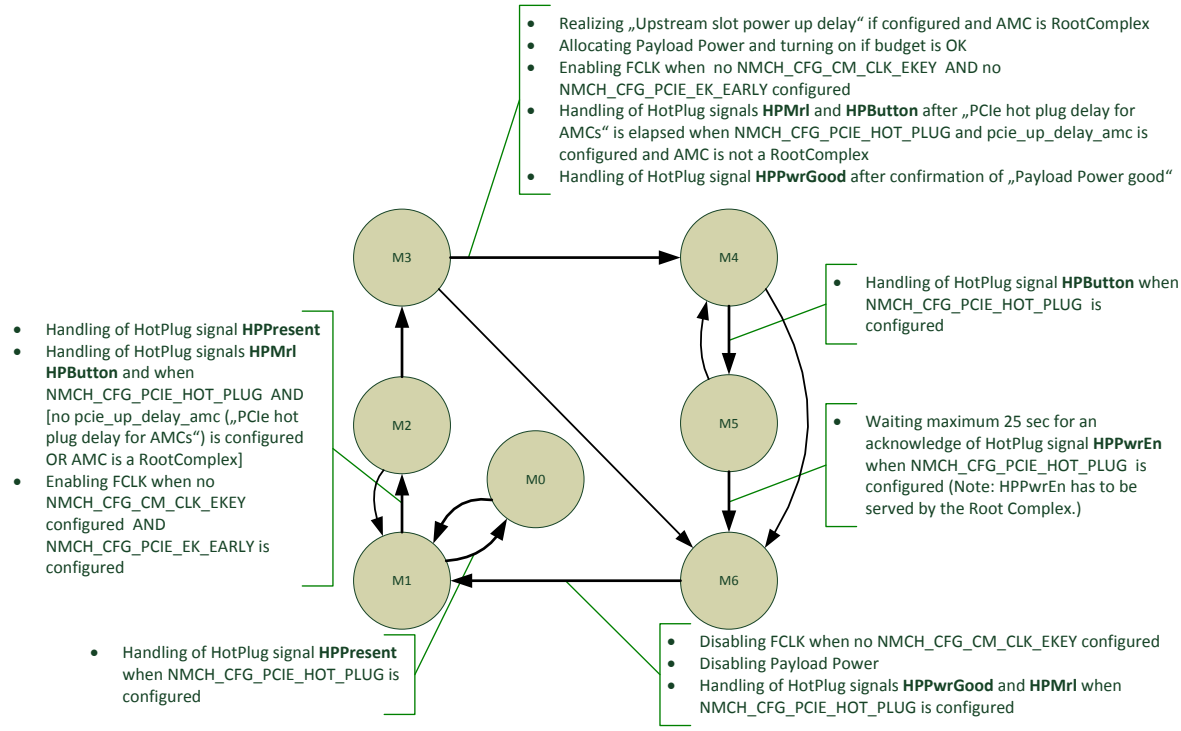


Figure 24: PCIe-HotPlug signal vs. FRU states

The hot plug signals are emulated by the MCH firmware on a Power Up reset always as a system with extracted AMCs even if the AMCs are inserted at that time. That means that the signals HPPresent, HPMrl, HPButton and HPPwrGood are served from MCH firmware like shown in Figure 17: PCIe-HotPlug signal vs. FRU states after Power Up (M0 → M1 → M2

→ M3 → M4). This is necessary for getting the Hot Plug Controller of the PCIe switch in the right state. The Root Complex has also to take care about the hot plug signals so that the bit settings in the register 80h Slot Status and Control of the downstream ports are set accurately for the Hot Plug Controller of the PCIe switch. Typically this is done from the BIOS on a Root Complex. In case that the BIOS doesn't take care of the hot plug signals correctly it might be helpful to set the PCIe configuration parameter "PCIe hot plug delay for AMCs". The delay time should be set to a value which makes sure that the HPMrl signal from the PCIe end points are signaled after the operation system of the Root Complex can take care on the handling of the hot plug signals instead of the BIOS. In this case the operation system of the Root Complex sets up the Hot Plug Controller of the PCIe switch accurately.

Document's History

Version	Date	Description	Author
1.0	10.6.2006	Initial Version based on V1.6 of the Quick Install Guide Extracted Firmware Release Info into separate document	hl
1.1	26.07.2007	Updated chapters about management interface and command line interface for MCH firmware V2.0, added glossary, list of figures, list of tables	cg
1.2	17.08.2007 20.08.2007	Added SEL configuration bit description. Reworked.	Cg hl
1.3	06.09.2007	Added Gigabit Ethernet Switch configuration bit description	ww
1.4	26.09.2007	In GeSwitchFlags added bit description for enabling 802.1Q VLAN support	ww
1.5	08.10.2007	Added 'ping' and 'route' CLI command's and IP configuration description.	Cg
1.6	10.10.2007 16.11.2007	Removed Flag "FLAG_POWER_UP_DELAY" Added Flag "PCIE_HP_SUPPORT" Updated chapter 'Command line interface'	hl cg
1.7	20.11.2007 07.12.2007	Added RemoteAccessFlag in MCH configuration register overview table Added Chapter 'Local Shelf Manager' Added chapter 'Sensor Event Log'	hl hl cg
1.8	11.02.2008 25.02.2008 26.02.2008	Added pin out for the backplane connector of the 4th MCH tongue. Added chapter 'MCH access from a different subnetwork'. Updated chapter 'Command line interface'. New chapter about JAVA application NatView.	Ks cg ss
1.9	8.5.2008	Changed chapter MCH Configuration and adapted to new configuration structure	hl
1.10	20.06.2008 20.08.2008	Added description of PCIe clustering parameters in MCH configuration Added description for XAUI and SRIO Hub-Module Changed Mini-BNC to SMA connector	ww ks
1.11	10.09.2008	Added IP address source configuration for GbE interface in ch. 0. Updated CLI reference in ch. 8. Added note about IP address source configuration in Appendix B .	cg
1.12	09.02.2009	In ch. 0 added description of backward compatibility configuration and in ch. 5.4.2 removed description of OpenHPI compatibility because it is unused now.	Cg

1.13	10.02.2009	in chapter 5 added CLK and SRIO module configuration, in chapter 9 updated web configuration interface description added chapter **Fehler! Verweisquelle konnte nicht efunden werden.** for clock module configuration	ww
1.14	15.03.2009	Added MCH variants Added appendix private IPMI message definitions	hl
1.15	17.04.2009 20.4.2009	editorial changes updated OpenHPI and Command line interface chapters Added ipmitool call syntax chapter and appendix about clock and hub module register access via ipmitool. Added USB console connection description for MCH Gen3.	Cg hl
1.16	02.07.2009	Added note about GbE option of Gen2 MCH in chapter 3.1.1	ww
	03.07.2009	Added CLI command sw_mp in ch. 8 Added note about DHCP restrictions in ch. 0	ww
1.17	20.08.2009	Corrected some typos. Added clarifications related to MCH Gen3. Checked chapter about NATView, also added a new section about how to download the latest release. Update webserver description in ch. 9	hl ss ww
1.18	19.09.2009 12.10.2009 14.10.2009 07.12.2009 17.02.2010 09.03.2010 09.03.2010 25.03.2010 26.03.2010	Added remark regarding GbE Uplink port on NAT-MCH Gen3 in chapter 0 Added information for updating MCH by Web-Interface Added "update_mch" instructions for Firmware 2.7 or later Added ipmitool usage example to update firmware of a power-module Added missing NAT IPMI extensions Added mchcfg->clock e-keying Updated Webserver chapter 9 Update clock configuration chapter **Fehler!** **Verweisquelle konnte nicht gefunden werden.** Removed config flag "no ekey for PCIe Updated NATView related sections. Also added Appendix D.	ww cs cs cs cs ww ww hl ss
1.19	16.07.2010	Added note to only use tar-archives when using web-interface-update-function	cs

1.20	09.08.2010	In chapter 7.3.1.2 corrected FTP server link	Ww
1.21	10.08.2011 12.02.2013	Added Block Diagram Gen 3 / LC Added new company contact data Added restriction concerning Managed Ethernet Mode Minor changes, e.g. labeling Changed Boot String to new value 0xd0100008 Added telnet password	se hn
1.22	11.03.2013	Updated chapter "5.4.9 NTP Configuration [10]"	hn
1.23	30.04.2013 02.05.2013 16.05.2013	Added chapter "10.3 Clock Module Version 4.x" and "10.4 Clock Module Version Physics"; renamed former chapter "10.3 Common Commands for V2.x and V3.x" into "10.5 Common Commands for all Clock Modules" Added chapter "9.1 PCIe Virtual Switch Configuration" Added some missing descriptions in MCH configuration	hn hn ww
1.24	05.06.2013 25.06.2013 20.08.2013 28.08.2013 16.12.2013	In chapter 5.4.2 added description of "Send SEND_MSG confirmation to SMS" flag Added SMA connector description Typo correction, reworked wording Added hint the PCIe is not supported on AMC Ports 8-11 Corrected Tongue1/4 connector pin description	Ww Se Se Te te
1.25	4.02.2014 18.03.2014 6.05.2014 26.05.2014	Added Faceplate description for MCH Gen3 with PCIe Gen3 option Added chapter 2.5: MCH M4 Update of Table 2 (MCH Variants and Features) Added Block Diagram MCH M4 Added Appendix F Updating the Backplane EEPROM via MCH CLI Added chapter 10.4: Unmanaged AMC Module Configuration Added chapter 10.3: SRIO Module Configuration	se hn hn ww

1.26	15.08.2014	Merged all configuration script related descriptions into chapter 10	ww
	02.10.2014	Chapter 8: Removed CLI command route, arp (since Fw V2.15) Removed CLI command mac_amc (has not been supported since FW. V2.13) Renamed/reworked show_xlinkinfo=>show_xaui_links Renamed/reworked show_xmact=>show_xaui_mact Added description for CLI command: show_xaui_stat (since FW. V2.16) Added description for CLI command: show_gbe_links (since FW. V2.16)	al
	06.10.2014	Added description for CLI command: show_swp2p	al
	27.10.2014	Updated SSC information with "only applicable when the PCIe Hub module isn't assembled as option LOSC" Added description of PCIe configuration parameter 'no ekey' for PCIe Added chapter Clock Module Version USB	hn
	12.11.2014	Added description of CLI command "show_pwrconf"	hl
	14.11.2014	Added description of config flag "PM assignment strategy"	dh
	14.11.2014	Removed firmware update via CLI with TAR file	dh
	14.11.2014	Extended chapter 11	dh
	19.11.2014	Updated chapter 5.4.7 PCIe Switch Configuration [9] (optional) Updated chapter 9.1 PCIe Virtual Switch Configuration (optional)	hn
	20.11.2014	Added Appendix G PCIe Hot Plug Support (optional)	hn
	21.11.2014	Updated chapter 10.3 SRIO Module Configuration	hn
1.27	18.12.2014	Added new parameters in 10.3 SRIO Module Configuration	hn
1.28	07.04.2015	Added description of SSH interface	th

1.29	21.07.2015	Updated chapter "Webserver"	al
1.30	20.11.2015 08.01.2016	Added Watchdog Timer description Updated chapter 10.2.4.1Clock Output Configuration with Table 10-1 Clock Output Identifier Mapping in NATIVE-C5 or NATIVE-R5 chassis	dh hn
1.31	15.01.2016 28.04.2016 22.08.2016	Added chapter PCIe Module Configuration Updated chapter Clock Module Version Physics Updated chapter PCIe-Hub Module (optional) PCIe Virtual Switch Configuration (optional) with x8 PCIe Link Width configuration option	hn hn hn
1.32	03.03.2017	Added chapter "PCIe Error Counter" Added chapter "PCIe Link Status" Added chapter "SRIO GEN II - Status Menu" Added chapter "SRIO GEN II - Routing Table" Added chapter "SRIO GEN II - Error Rate" Added chapter "SRIO GEN II - Port Statistic"	al