What is New in MicroTCA.4.1?

Kay Rehlich, DESY MicroTCA Technology Lab

Auxiliary Backplane
Protective Covers
AMC / RTM Application Classes
PICMG® Specification MTCA.4.1 R1.0

Enhancements for MicroTCA.4

- Auxiliary Backplane for Rear Transition Modules (µRTMs & MCH RTM)
- Rear Power Modules (RPMs)
- MCH Management Support & Extended Rear Transition Module (MCH-RTM)
- AMC & RTM Protective Covers
- Applications Classes of µRTMs

November 1, 2016

168 Pages
MicroTCA.4: A Modular Crate System

- Space for a CPU & ….
- Redundant Fans
- Most cables from rear
- Space for …. MTCA.4.1
- 1 … 4 Power Supplies
- Filter

© Elma
MicroTCA Generations: MTCA.0 MTCA.4 MTCA.4.1

Front Side

PM PM AMC AMC AMC AMC AMC AMC AMC MCH PM PM

- Data
- Management
- Power
MicroTCA Generations: MTCA.0  **MTCA.4**  MTCA.4.1

Front Side

Rear Side

Data

Management

Power

Zone 3 Connection

PM  PM

AMC  AMC  AMC  AMC  AMC  AMC  AMC  AMC  AMC  AMC

mRTM  mRTM  mRTM  mRTM  mRTM  mRTM  mRTM  mRTM  mRTM

MCH-RTM  MCH  PM  PM


Kay Rehlich, DESY
MicroTCA Generations: MTCA.0 MTCA.4 MTCA.4.1

Front Side

Rear Side
RPM: Rear Power Module
Provides:
• 3.3 V management power to eRTM
• 12 V to eRTM
• Positive variable Voltage to RTM
• Negative variable Voltage to RTM
Managed by MCH via MCH-RTM
**MicroTCA.4.1 Rear Power Module: Variable Voltage**

**Example**

- **Requested voltage:** 6 .. 11 V, 5 .. 10 V, 4 .. 13 V, 5 .. 14 V
- **Provided voltage:** 15 V, 0 V
- **Calculated upper bound:** 15 V
- **Calculated lower bound:** 0 V
- **Target voltage:**

```plaintext
[6 .. 11 V, 5 .. 10 V, 4 .. 13 V, 5 .. 14 V]
```
**MicroTCA.4.1 MCH Rear Transition Module**

- **MCH-RTM**: MCH Rear Transition Module
  - Zone 3 plug is defined in MTCA.4.1
  - The MCH is the manager, the RTM links the $\mu$RTM Backplane to the MCH
  - Low power eRTMs can be powered from the MCH
MicroTCA.4.1 MCH Rear Transition Module

- Up to 16 additional fat-pipe for e.g. a CPU on MCH-RTM
- Management of a second µRTM backplane
**MicroTCA.4.1 MCH Rear Transition Module**

### Zone 3 connector definition:
- 16 fat-pipe (PCIe) lanes
- 2 SATA links
- 2 Ethernet ports
- Management of rear Power Modules
- Management of rear backplane EPROM
- Management of rear eRTMs
- Management of MCH-RTM
- Power

### Zone 2 connector definition:
- Management of rear Power Modules
- Management of rear backplane EPROM
- Management & power for rear eRTMs
- 6 LVDS to 3 eRTMs (user defined)
MicroTCA.4.1 MCH-RTM: 3 * 2 Fat-Pipes

Supported Protocols:
- PCI Express
- PCI Express Advanced Switching
- Ethernet 1000Base-X
- Serial RapidIO
- Serial Peripheral Interface
- Universal Asynchronous Receiver-Transmitter
- Universal Synchronous Receiver-Transmitter
- Ethernet 100Base-T
MicroTCA.4.1 Extended Rear Transition Module

- **eRTM 13 or 2nd MCH-RTM:** Power from RPM
- **eRTM 14:** Power from RPM
- **eRTM 15:** Power from MCH-RTM or RPM. Example: RF and Clock distribution
- **1 or 2 RPM:**
  - Power to:
    - eRTM
    - $\mu$RTM
    - MCH-RTM
  - Optional
Example $\mu$RTM Backplane

- **eRTM 15**: Power from MCH-RTM or RPM
  Example: RF and Clock distribution

- **eRTM 14**: Power from RPM

- **eRTM 13** or 2nd MCH-RTM:
  Power from RPM

- **1 or 2 RPM**: Power to:
  - eRTM
  - $\mu$RTM
  - MCH-RTM
µRTM Backplane

With one MCH-RTM

User definable plugs

Example: Multi-Coax plugs for very low jitter RF and clock distribution

With two MCH-RTM
Position of the $\mu$RTM Backplane

• The distance of the plugs on both backplanes are fixed
  • Independent of the backplane thickness
MicroTCA.4.1 AMC Protective Covers

• Mechanical protection of components
• A bit of shielding (grounded stainless steel with insulating foil)
MicroTCA.4.1 RTM Protective Covers

Side 1

Zone 2 Connector (optional)
MicroTCA.4.1 AMC Protective Covers

Keying block

2.7Ø Thru hole for standoff with screw, connected to chassis GND
MicroTCA.4.1 Better Modularity of RTMs: Classes

- Definition of AMC / RTM compatibility by classes and sub-classes
- AMC and RTM both have FRU “Classification Records”
- One module can define compatibility with several classes
- Pin and Signal description of the Zone 3 plug
- Compatibility means: can be switched on
- Compatibility does NOT mean that all features are implemented

https://mtca.desy.de/resources/zone_3_recommendation/index_eng.html
## AMC - RTM Compatibility Check: Classes

### AMC

<table>
<thead>
<tr>
<th>Class Descr. Count</th>
<th>Class Type: A or D</th>
<th>Class ID major</th>
<th>Class ID minor</th>
</tr>
</thead>
</table>

### RTM

<table>
<thead>
<tr>
<th>Compatibility Record</th>
<th>Class Descr. Count</th>
<th>Class Type: A or D</th>
<th>Class ID major</th>
<th>Class ID minor</th>
</tr>
</thead>
</table>

AMC is compatible with RTM if ≥ 1 record is matching.

---

**MicroTCA.4.1**
MicroTCA.4.1 AMC - RTM: Class Definition

• Analog Class A1.1
  ‣ 10 differential analog inputs, AC
  ‣ 10 differential analog inputs, DC
  ‣ 5 differential analog outputs, DC
  ‣ Clocks: 6 input, AC
  ‣ 9 differential digital IO
  ‣ 2 high speed links optional

• Analog Class A2.1
  ‣ 32 differential analog inputs, DC
  ‣ 4 differential analog outputs, DC
  ‣ Clocks: 2 input (AC), 1 output (LVDS)
  ‣ 6 differential digital IO (LVDS)
Summary

MicroTCA.4.1

✓ First standard of a rear backplane with full management
✓ Allows very compact and modular designs
✓ Class concept for a modular AMC/RTM compatibility
**Standard Hardware API Design Guide**

- **User Application**
  - Logical ADC 1
  - Logical ADC 2
  - ..
  - Address Mapping

- **CPU**

- **FPGA**
  - Address Mapping
  - Logical ADC 1
  - Logical ADC 2
  - ..
  - User FPGA Code

---

Get Configuration

---

SHAPI

• Self-describing communication: CPU <-> FPGA
  ‣ Decoupling of application programs from FPGA code
  ‣ No address mapping file required
  ‣ Re-usable FPGA and application modules
• One Device header describes the FPGA and board hardware
  ‣ Provides the main interrupt registers (vectored interrupt preferred)
  ‣ Links to the first module of the FPGA
• Module headers are describing the function blocks (e.g. ADCs, DACs, …)
  ‣ A module has a standard part (ID, capabilities, …) and a function specific part
• DMA is implemented by a special DMA Module
SHAPI: FPGA Device is Composed of Module

Addr: 0x00

- **SHAPI Device**; version
  - Next addr
  - Firmware ID, version
  - Device capabilities
  - Interrupts

SHAPI Module; version
- Next addr
- Firmware ID, version
- Device capabilities
- Interrupts

SHAPI Module; version
- Next addr: 0 = last
- Firmware ID, version
- Device capabilities
- Interrupts
SHAPI: Standard Device Identification

### Capabilities:
- Full reset available
- Soft reset available
- Has RTM detection
- Endianness available

### Status:
- Doing Full reset routine
- Doing Soft reset routine
- RTM present

### Control:
- Do Full reset
- Do Soft reset
- Use big-endian format

---

<table>
<thead>
<tr>
<th>Addr</th>
<th>R/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>ro</td>
</tr>
<tr>
<td>0x04</td>
<td>ro</td>
</tr>
<tr>
<td>0x08</td>
<td>ro</td>
</tr>
<tr>
<td>0x0C</td>
<td>ro</td>
</tr>
<tr>
<td>0x10</td>
<td>ro</td>
</tr>
<tr>
<td>0x14</td>
<td>ro</td>
</tr>
<tr>
<td>0x18</td>
<td>ro</td>
</tr>
<tr>
<td>0x1C</td>
<td>ro</td>
</tr>
<tr>
<td>0x20</td>
<td>ro</td>
</tr>
<tr>
<td>0x24</td>
<td>ro</td>
</tr>
<tr>
<td>0x28</td>
<td>ro</td>
</tr>
<tr>
<td>0x2C</td>
<td>rw</td>
</tr>
<tr>
<td>0x30</td>
<td>rw</td>
</tr>
<tr>
<td>0x34</td>
<td>ro</td>
</tr>
<tr>
<td>0x38</td>
<td>ro</td>
</tr>
<tr>
<td>0x3C</td>
<td>rw</td>
</tr>
</tbody>
</table>
### SHAPI: Device Interrupt Registers

<table>
<thead>
<tr>
<th>Field</th>
<th>Addr</th>
<th>R/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magic = 0X5348</td>
<td>0x00</td>
<td>ro</td>
</tr>
<tr>
<td>SHAPI Version</td>
<td>0x00</td>
<td>ro</td>
</tr>
<tr>
<td>First Module Address</td>
<td>0x04</td>
<td>ro</td>
</tr>
<tr>
<td>Hardware ID</td>
<td>0x08</td>
<td>ro</td>
</tr>
<tr>
<td>HW Vendor ID</td>
<td>0x08</td>
<td>ro</td>
</tr>
<tr>
<td>Device FW ID</td>
<td>0x0C</td>
<td>ro</td>
</tr>
<tr>
<td>Device Vendor ID</td>
<td>0x0C</td>
<td>ro</td>
</tr>
<tr>
<td>Firmware Version</td>
<td>0x10</td>
<td>ro</td>
</tr>
<tr>
<td>Firmware Timestamp</td>
<td>0x14</td>
<td>ro</td>
</tr>
<tr>
<td>Firmware Name</td>
<td>0x18</td>
<td>ro</td>
</tr>
<tr>
<td>Device Capabilities</td>
<td>0x24</td>
<td>ro</td>
</tr>
<tr>
<td>Device Status</td>
<td>0x28</td>
<td>ro</td>
</tr>
<tr>
<td>Device Control</td>
<td>0x2C</td>
<td>rw</td>
</tr>
<tr>
<td>Interrupt Mask</td>
<td>0x30</td>
<td>rw</td>
</tr>
<tr>
<td>Interrupt Flag</td>
<td>0x34</td>
<td>ro</td>
</tr>
<tr>
<td>Interrupt Active</td>
<td>0x38</td>
<td>ro</td>
</tr>
<tr>
<td>Scratch Registers</td>
<td>0x3C</td>
<td>rw</td>
</tr>
</tbody>
</table>

#### Interrupt Mask:
- 32 bits
- 1 = interrupt **enabled** for the device assigned to this bit

#### Interrupt Flag:
- 32 bits
- 1 = interrupt was **raised**
- Is **masked**

#### Interrupt Active:
- 32 bits
- 1 = interrupt was **raised**
- Is **not masked**
# SHAPI: Standard Module Identification

## SHAPI Overview

- **Vendor ID**: Can be used for plug-in modules
- **Firmware ID**: Defines the routine to handle this module, e.g. a certain ADC type
- **Version**: Major (SW compatibility), minor, patch
- **Capabilities, Status, Control**: Full or soft reset (available, status, ctrl), RTM required, Single or multiple interrupts
- **Interrupts**: ID: defines which device level interrupts can be raised by this module (one bit), Clear, Mask, Flag, Active: to reset, enable, IRQ & Mask, IRQ

## Module Identification & Control

<table>
<thead>
<tr>
<th>Field</th>
<th>Addr</th>
<th>R/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magic = 0x534D</td>
<td>0x00</td>
<td>ro</td>
</tr>
<tr>
<td>Next Module Address</td>
<td>0x04</td>
<td>ro</td>
</tr>
<tr>
<td>Firmware ID</td>
<td>0x08</td>
<td>ro</td>
</tr>
<tr>
<td>Module Version</td>
<td>0x0C</td>
<td>ro</td>
</tr>
<tr>
<td>Module Name: std_dma</td>
<td>0x10, 0x14</td>
<td>ro</td>
</tr>
<tr>
<td>Module Capabilities</td>
<td>0x18</td>
<td>ro</td>
</tr>
<tr>
<td>Module Status</td>
<td>0x1C</td>
<td>ro</td>
</tr>
<tr>
<td>Module Control</td>
<td>0x20</td>
<td>rw</td>
</tr>
<tr>
<td>Interrupt ID</td>
<td>0x24</td>
<td>ro</td>
</tr>
<tr>
<td>Interrupt Flag Clear</td>
<td>0x28</td>
<td>rw</td>
</tr>
<tr>
<td>Interrupt Mask</td>
<td>0x2C</td>
<td>rw</td>
</tr>
<tr>
<td>Interrupt Flag</td>
<td>0x30</td>
<td>ro</td>
</tr>
<tr>
<td>Interrupt Active</td>
<td>0x34</td>
<td>ro</td>
</tr>
<tr>
<td>N Module Specific Registers</td>
<td>0x38 …</td>
<td></td>
</tr>
</tbody>
</table>

---


Kay Rehlich, DESY
### SHAPI: DMA Standard Module

<table>
<thead>
<tr>
<th>Address (Addr)</th>
<th>R/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>ro</td>
</tr>
<tr>
<td>0x04</td>
<td>ro</td>
</tr>
<tr>
<td>0x08</td>
<td>ro</td>
</tr>
<tr>
<td>0x1C</td>
<td>ro</td>
</tr>
<tr>
<td>0x20</td>
<td>rw</td>
</tr>
<tr>
<td>0x24</td>
<td>ro</td>
</tr>
<tr>
<td>0x28</td>
<td>rw</td>
</tr>
<tr>
<td>0x2C</td>
<td>rw</td>
</tr>
<tr>
<td>0x30</td>
<td>ro</td>
</tr>
<tr>
<td>0x34</td>
<td>ro</td>
</tr>
<tr>
<td>0x38</td>
<td>ro</td>
</tr>
<tr>
<td>0x3C</td>
<td>ro</td>
</tr>
<tr>
<td>0x40</td>
<td>rw</td>
</tr>
<tr>
<td>0x44, 0x48</td>
<td>rw</td>
</tr>
<tr>
<td>0x4C, 0x50</td>
<td>rw</td>
</tr>
<tr>
<td>0x54</td>
<td>rw</td>
</tr>
<tr>
<td>0x58</td>
<td>ro</td>
</tr>
<tr>
<td>0x5C, 0x60</td>
<td>rw</td>
</tr>
<tr>
<td>0x64,0x68</td>
<td>rw</td>
</tr>
<tr>
<td>0x6C</td>
<td>rw</td>
</tr>
<tr>
<td>0x70</td>
<td>ro</td>
</tr>
</tbody>
</table>

**Vendor ID = 0**

**Firmware ID = 1**

Defines a DMA Module

### DMA Capabilities:
- Simultaneous read/write transfer
- RTM required
- Single or multiple interrupts
- To device DMA supported
- From device DMA supported
- Endianness data swap supported

### DMA:
- Destination Address (e.g. CPU)
- Source Address (e.g. board memory)
- Requested bytes to be transferred
- Bytes transferred in last DMA
SHAPI: Interrupts

- Module IRQs are mapped to the Device by the Interrupt ID register
- Device IRQs are pointing to the CPU vectored interrupts
- The CPU can implement an interrupt routine per Module
SHAPI & Linux Driver Repository

• GitHub as an open repository to exchange MicroTCA code
  ‣ Linux driver: hot-swap and universal
  ‣ SHAPI driver coming soon
  ‣ SHAPI FPGA test project coming soon

https://github.com/MicroTCA