# Do’s and Don’ts of MicroTCA System Design.

10th MicroTCA Workshop

Cagil Gumus (CJ)
Hamburg, 7 December 2021

HELMHOLTZ RESEARCH FOR GRAND CHALLENGES

DESY.

# Motivation

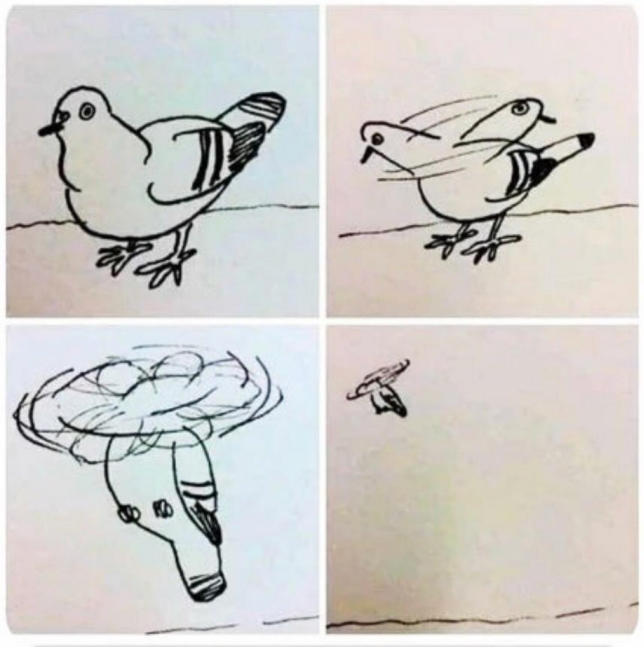
**MicroTCA can get very simple and very complicated.**

**Your application requirements can change the MicroTCA system significantly.**

- Many different applications using the same standard →
  - The standard is quite flexible →
    - System engineering becomes less straight forward
- For the newcomers, the task of assembling a new system from scratch can be daunting.

**This tutorial will show some of the critical elements of MicroTCA systems that effect design decisions.**

Credit: r/birdsarentreal



*It works != Best solution*

# Choose the right MTCA Crate

# Choose the right MTCA Crate

## The trade off between functionality – redundancy – reliability

***Questions to ask:***

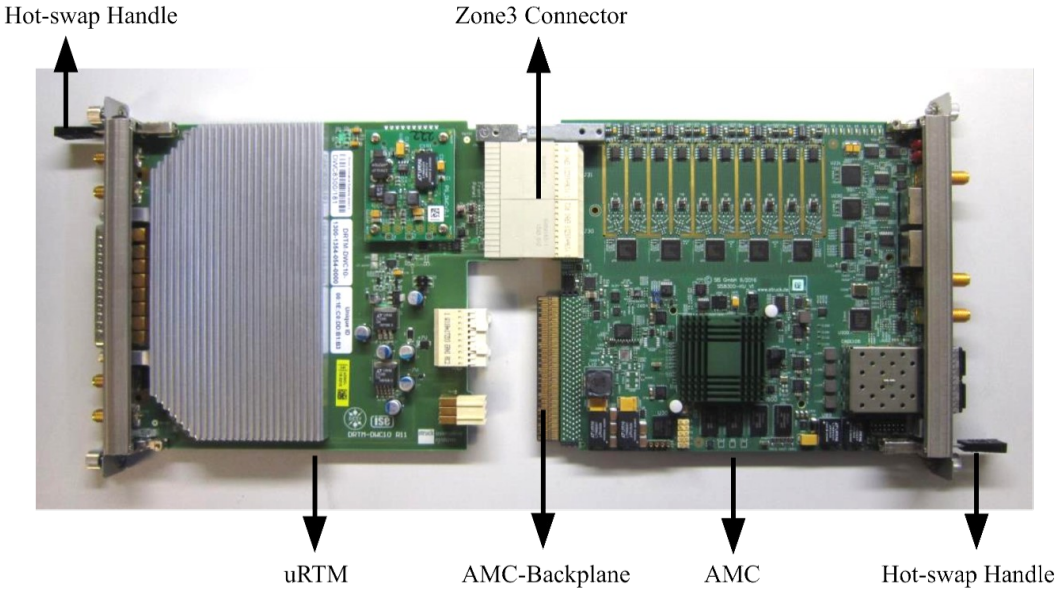
- How many AMC cards do you need?
- How about redundancy? (Power Module, MCH …)
- What kind of AMC cards?
- RTM cards necessary?
- RF Backplane Necessary?
- How should be the AMC Backplane configuration?
  - Fat Pipe Configuration (PCIe, 10/40 GigE, SRIO ..)
  - Point-to-Point Links
  - SATA
  - JTAG on Backplane?

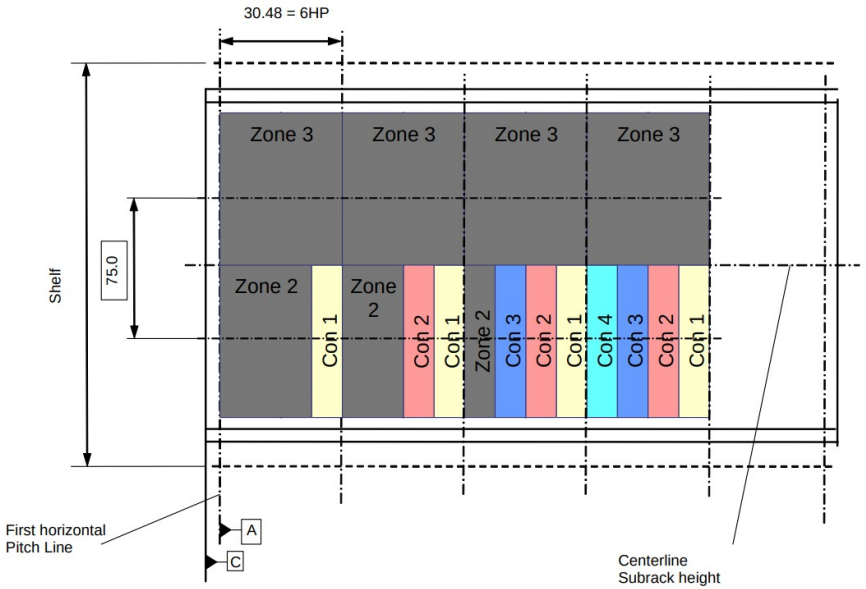
# Choose the right AMC+RTM Pair

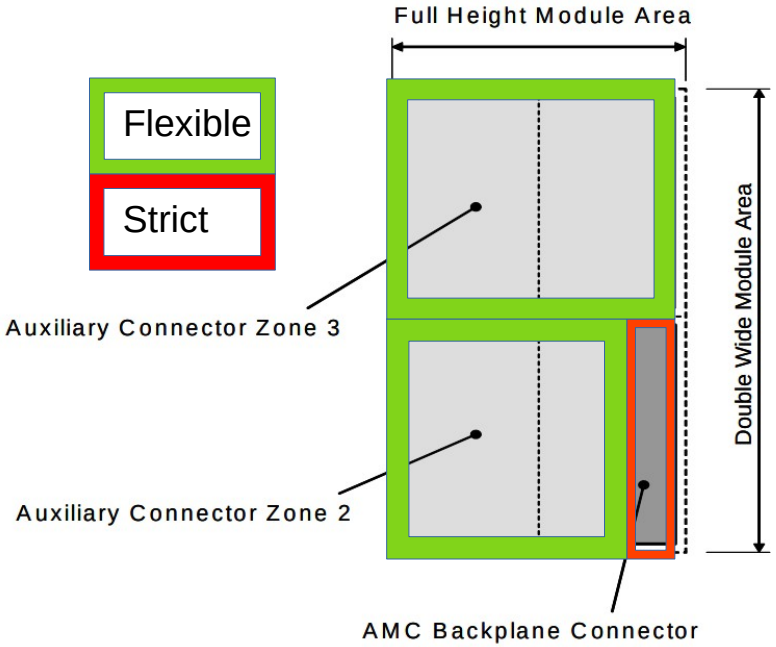
# Choosing the right AMC + RTM

## Importance of Zone2/3 Connectivity

- Mostly : AMC → COTS
- RTM → In-house development or COTS
- The MicroTCA.4 Standard does **not** dictates how Zone2 and Zone3 connector should be.
- There are recommondations done by companies/facilities.
- The interoperability might be an issue.



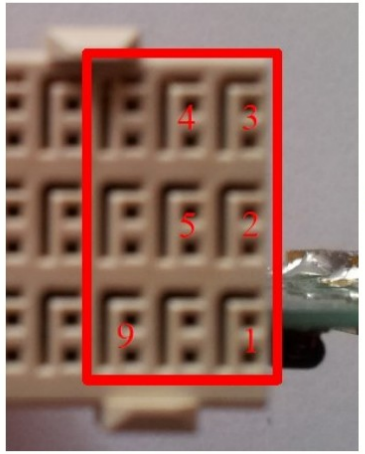



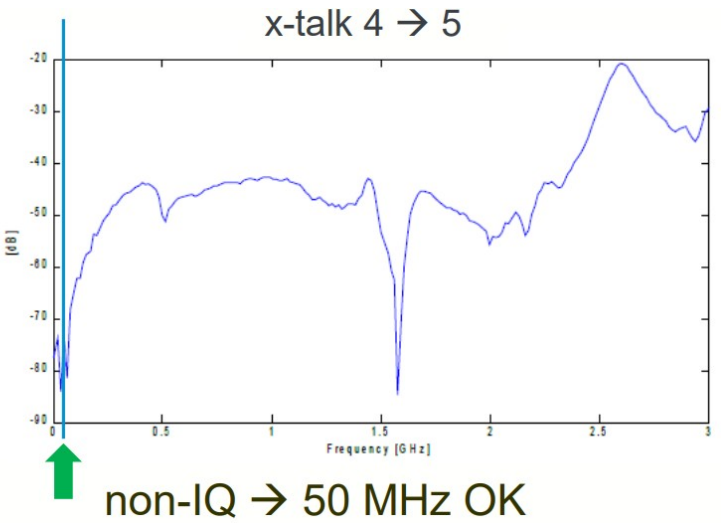



# Choosing the right AMC + RTM

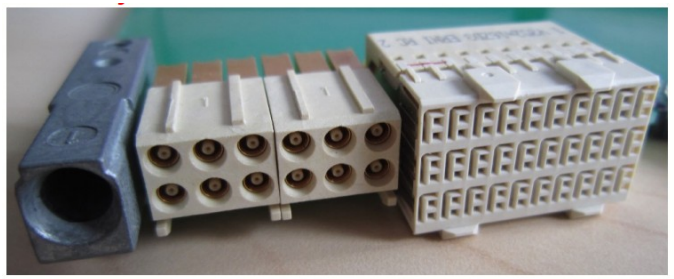
## Analog signal performance of Zone3

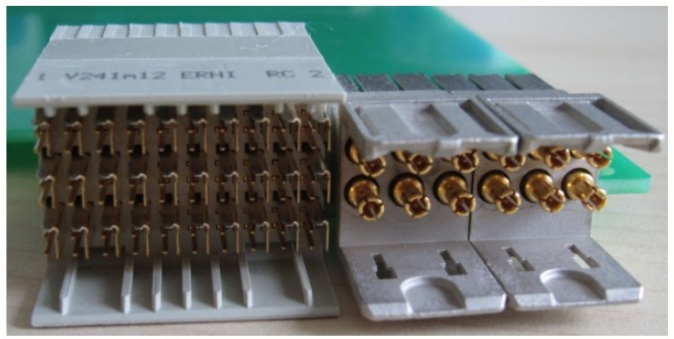
*A new Zone 3 Class for RF Signals up to 3 GHz in MicroTCA.4 Johannes Zink, MTCA Workshop 2019*




- Critical Question: How do I transfer analog signals inside the MicroTCA environment?
  - Direct injection from front panel of AMC
  - Over Zone3
- Analog signal transfer over Zone3 can be limited in terms of maximum frequency >200MHz is problematic for LLRF applications
- Solution: New connector: COAXIPACK 2 from Radiall
  - Upto 3GHz
-

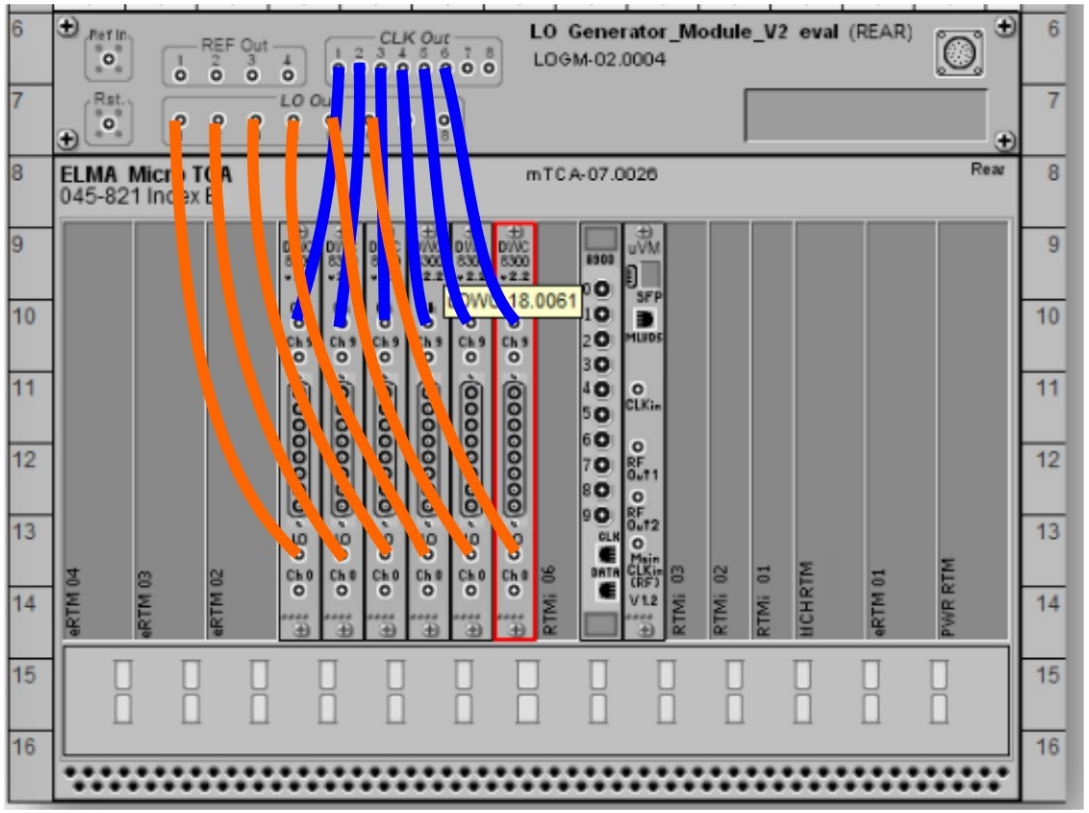




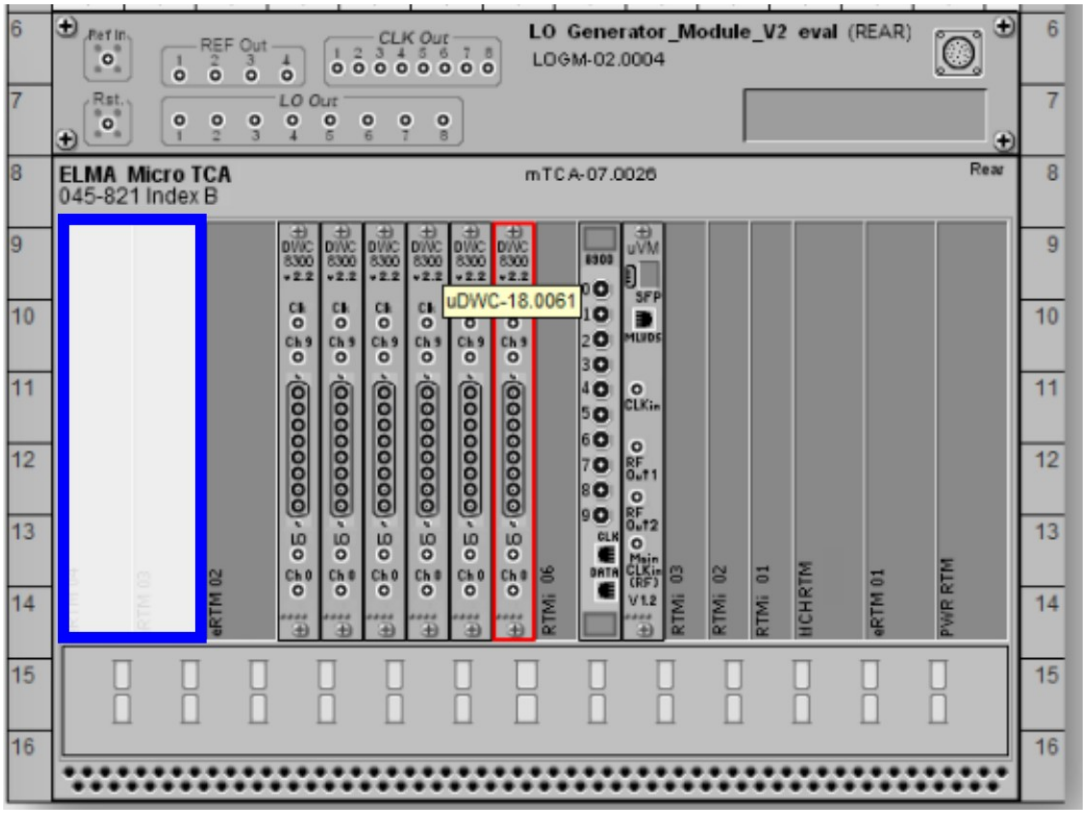
# RF Backplane (MicroTCA.4.1)

## Motivation: Getting rid of spaghetti, better management for analog signal distribution

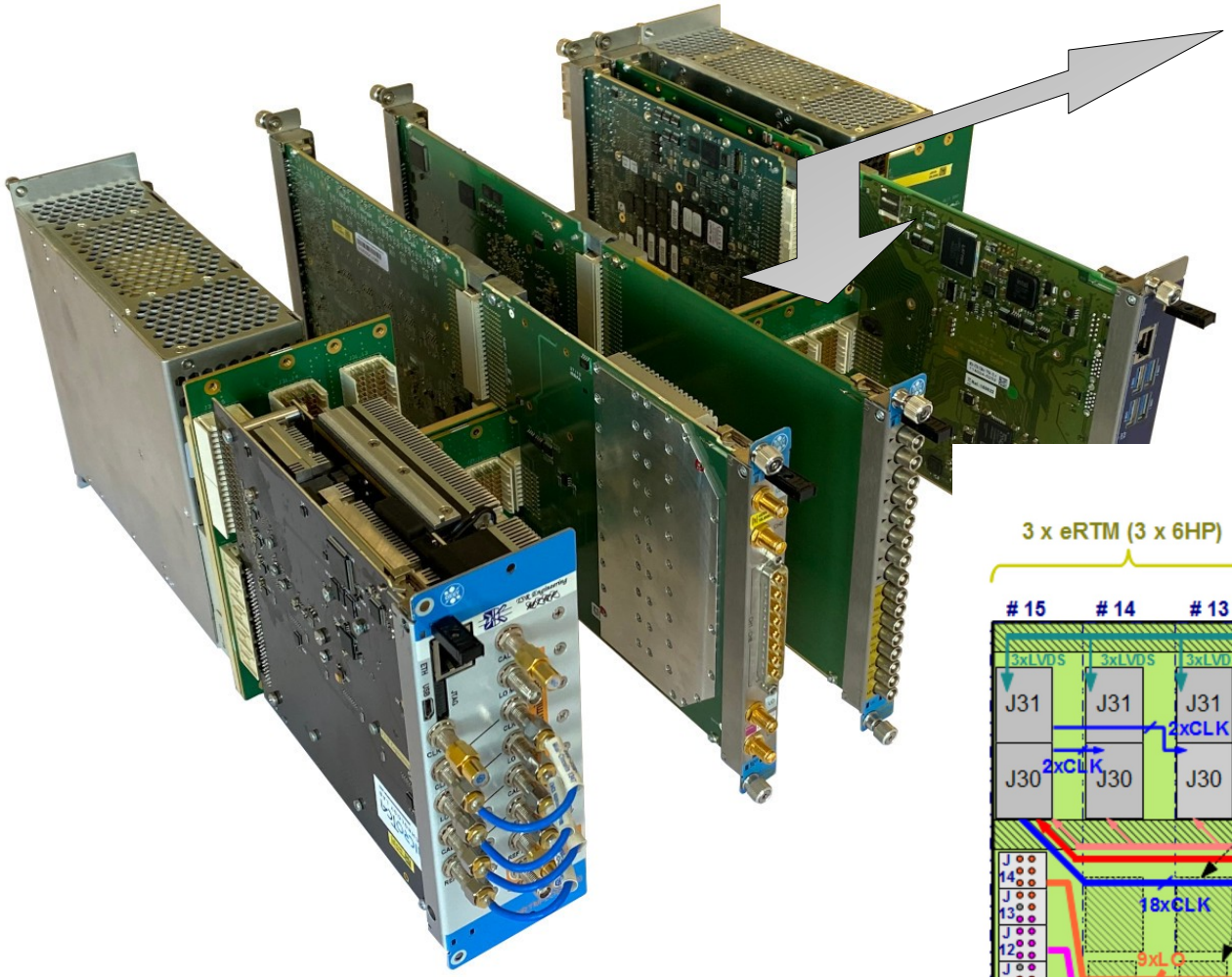
Before

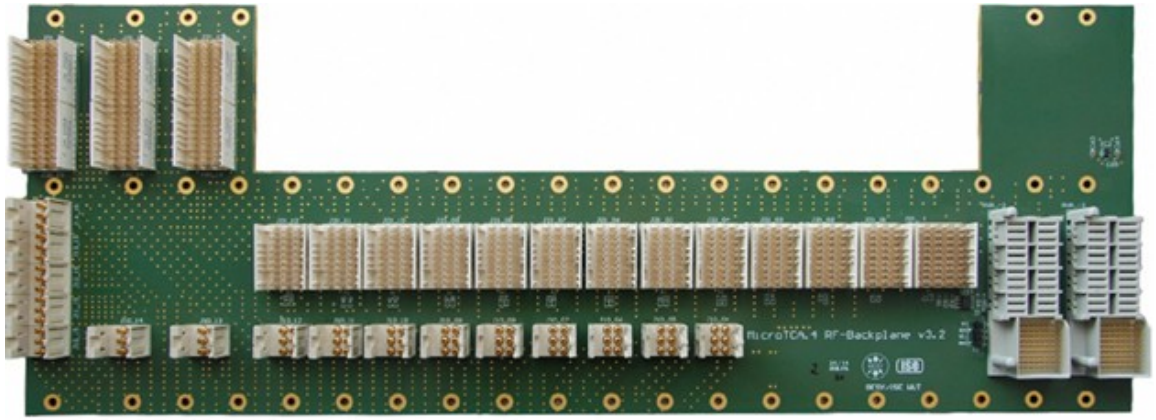


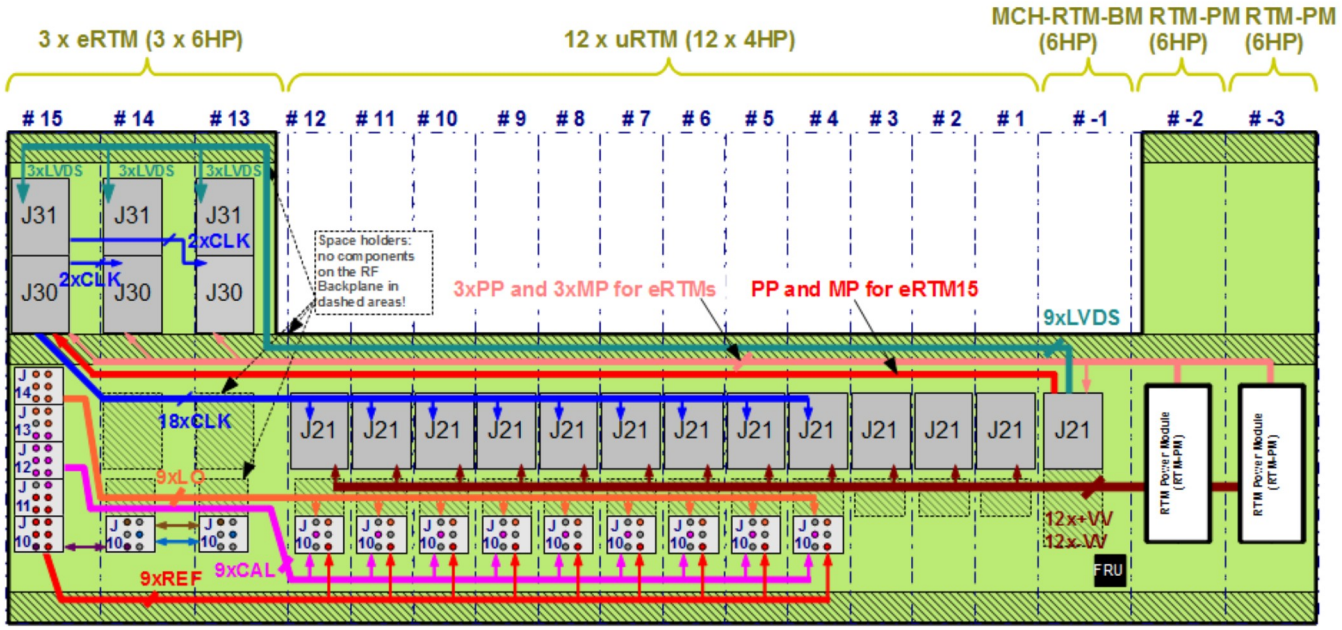

After



# RF Backplane





# Know your AMC-Backplane

# Know your AMC Backplane

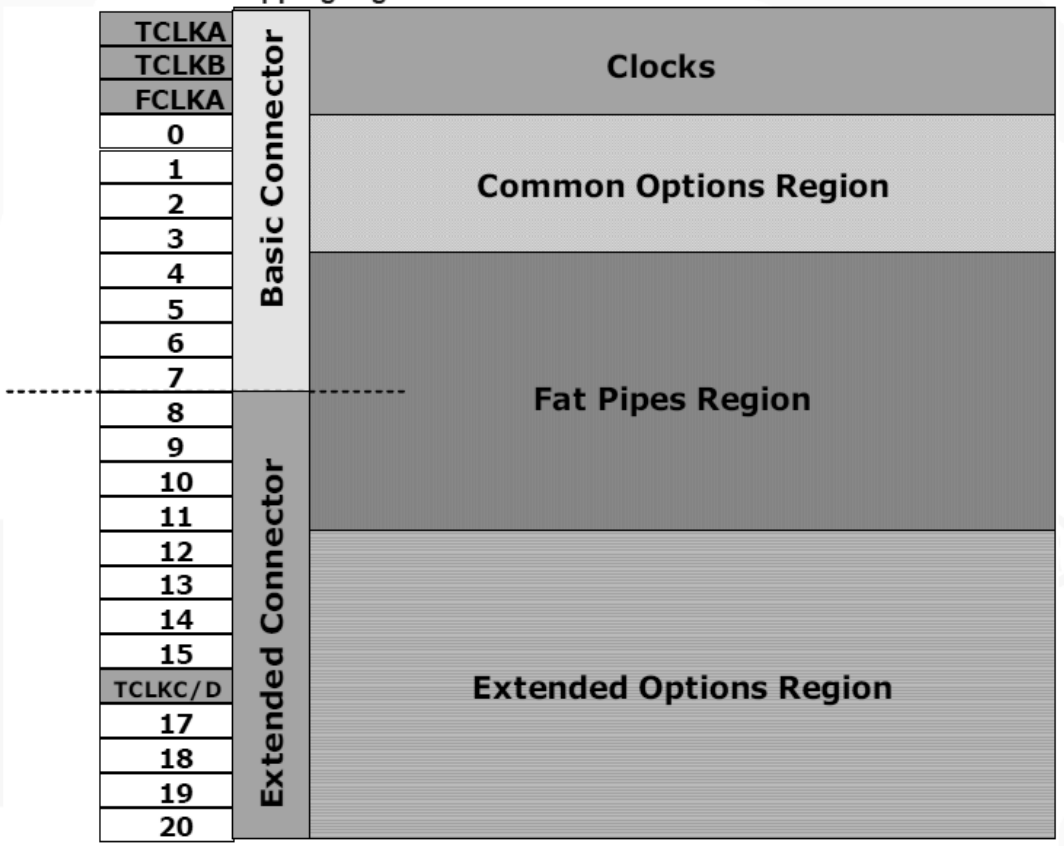
## Which ports to use on your application?

Protocols on the AMC backplane

- IPMI (Management)
- Gigabit Ethernet (Ports 0-1)
- SATA* (Ports 2-3)
- Fat Pipe + Extended Fat Pipe* (Ports 4-11)
  - PCIe
  - SRIO
  - 10/40 GbE
- Point-to-Point Links* (Ports 12-15)
- MVLDS* (Ports 17-20)
- Clocks* (TCLKA,TCLKB TCLKC,FCLK)
- JTAG*

- * → Changes depending on the crate/application

Figure 6-11 AMC Port mapping regions




Pro Tip:

Don’t know how your crate backplane looks like?

Backplane configuration is stored on the Carrier FRU EEPROM (FRU ID: 253)

NAT MCH: ‘show_fruinfo 253’

# Know your AMC Backplane

## Eg: PCIe

MicroTCA Crate can offer PCIe lanes in different ways:

---

Ports 4-7 (x4) → MCH #1

Ports 8-11(x4) → MCH #2

---

Ports 4-11(x8) → MCH #1

---

**Critical Question #1:**
How much bandwidth/latency does your application require?

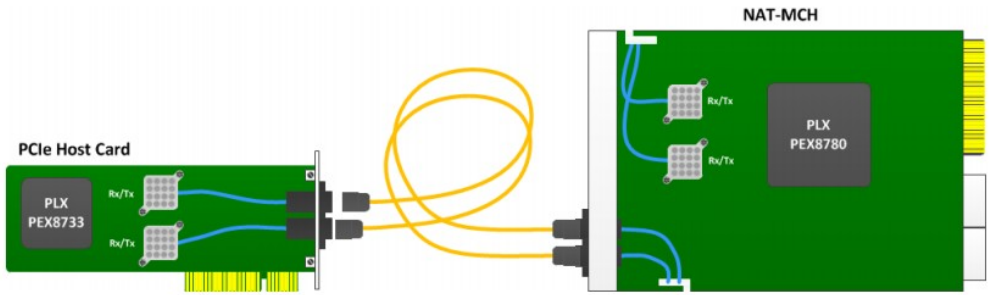
**Critical Question #2:**
Does my MicroTCA crate satisfy the answer to Critical Question #1?

PCI Express link performance[46][47]

| | Version | Intro-duced | Line code | Transfer rate[i][ii] | Throughput[i][iii] | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | ×1 | ×2 | ×4 | ×8 | ×16 |
| | 1.0 | 2003 | 8b/10b | 2.5 GT/s | 0.250 GB/s | 0.500 GB/s | 1.000 GB/s | 2.000 GB/s | 4.000 GB/s |
| | 2.0 | 2007 | 8b/10b | 5.0 GT/s | 0.500 GB/s | 1.000 GB/s | 2.000 GB/s | 4.000 GB/s | 8.000 GB/s |
| Now | 3.0 | 2010 | 128b/130b | 8.0 GT/s | 0.985 GB/s | 1.969 GB/s | 3.938 GB/s | 7.877 GB/s | 15.754 GB/s |
| | 4.0 | 2017 | 128b/130b | 16.0 GT/s | 1.969 GB/s | 3.938 GB/s | 7.877 GB/s | 15.754 GB/s | 31.508 GB/s |
| Future | 5.0 | 2019 | 128b/130b | 32.0 GT/s | 3.938 GB/s | 7.877 GB/s | 15.754 GB/s | 31.508 GB/s | 63.015 GB/s |
| | 6.0 (planned) | 2021 | 128b/130b + PAM-4 + ECC | 64.0 GT/s | 7.877 GB/s | 15.754 GB/s | 31.508 GB/s | 63.015 GB/s | 126.031 GB/s |

# PCIe Root Complex outside of the crate

## Suffering from weak CPU-AMC? Here is your solution




Needed Parts:

- 4 x Finisar BOA
- 4 x Pig Tail
- 4 x Face Plate Adapter
- 2 x Patch Cord 5m
- Resulting Costs for a PCIe

GenIII x16 Uplink Connection:

Effects MCH selection!

Pros:

- Cheaper & Poweful PC outside of 80W limitation
- Many choices in the industry for parts
- Many more PCIe slots available on the motherboard for more cards

Cons:

- CPU is not managed by MCH
- Boot sequence of crate and PC has to be done properly

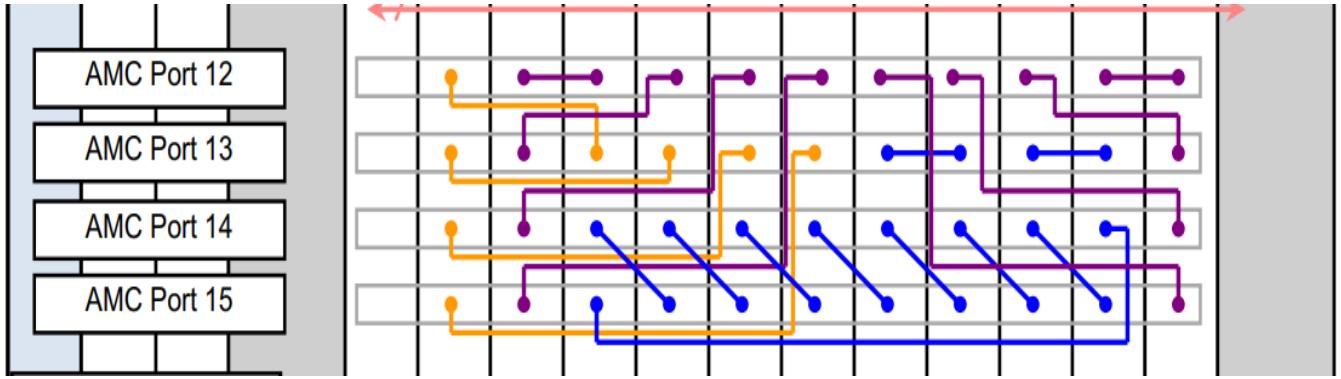
# Know your AMC Backplane

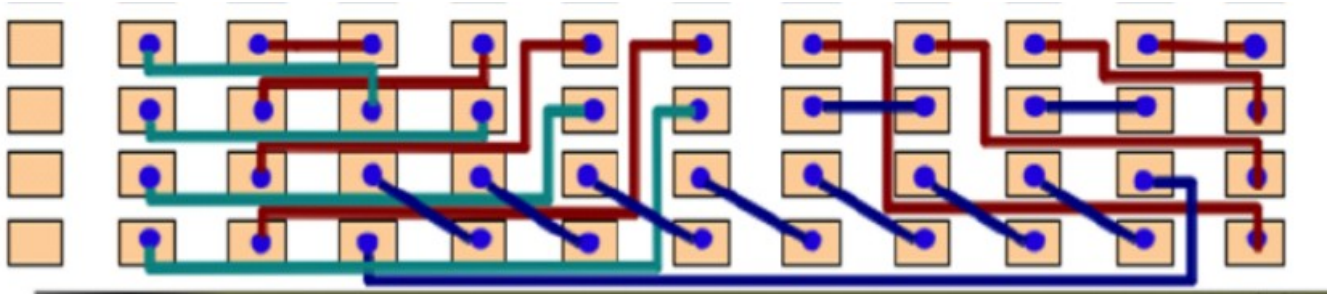
## Point to Point Links

Point to Point links offer direct communication from FPGA to FPGA.

Used for data aggregation / fast feedback between boards

These lines are ‘hard wired’. Double check the connectivity before ordering.



# Know your AMC Backplane

## Examples of P2P Links

Use Case Example:

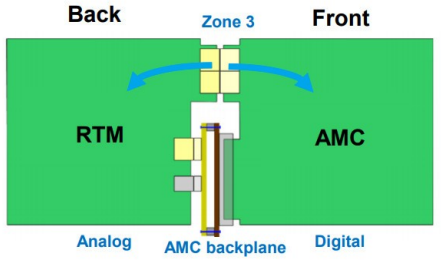
Data aggregation on point-to-point links on Europan-XFEL LLRF Crates:

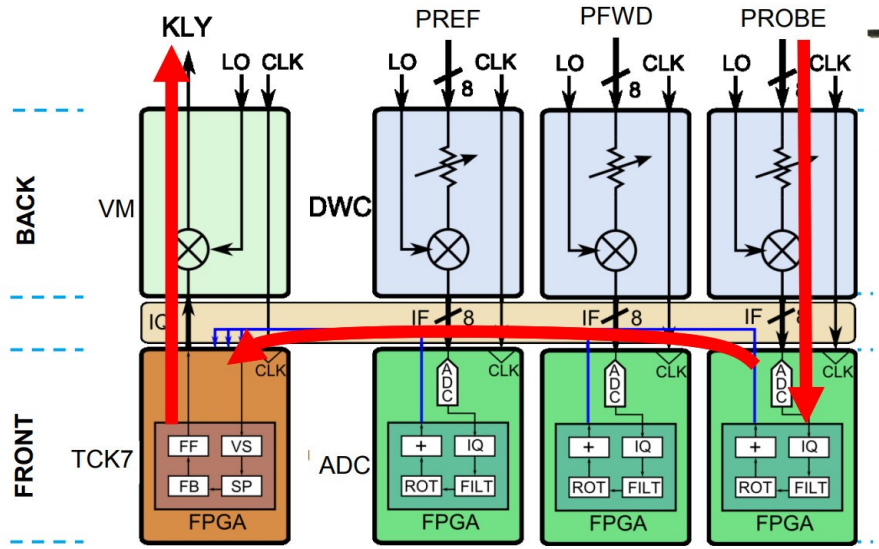
Probe + Forward + Reflected signals of 16 cavities gets send to main controller board.

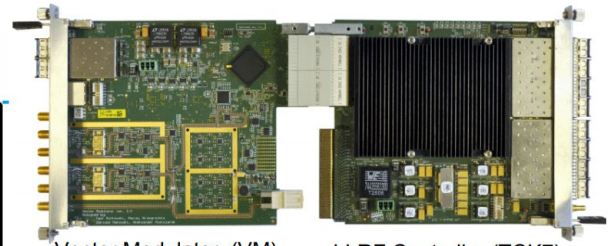
Some numbers:
6.25Gbps link rate
Sending 11x32 bits payload packet
End to End latency: ~344ns

Higher data rates with fully occupied crates harder to achieve because of big EMI issue

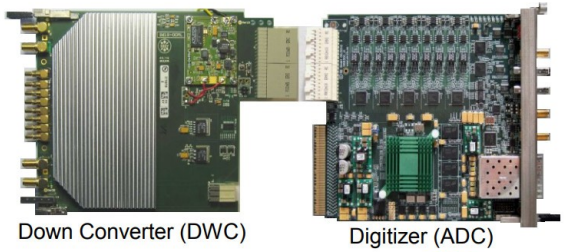
- AMC: Advanced Mezzanine Card
- RTM: Rear Transition Module
- 12 slots, hot swap
- Redundant power supply





Vector Modulator (VM)


LLRF Controller (TCK7)

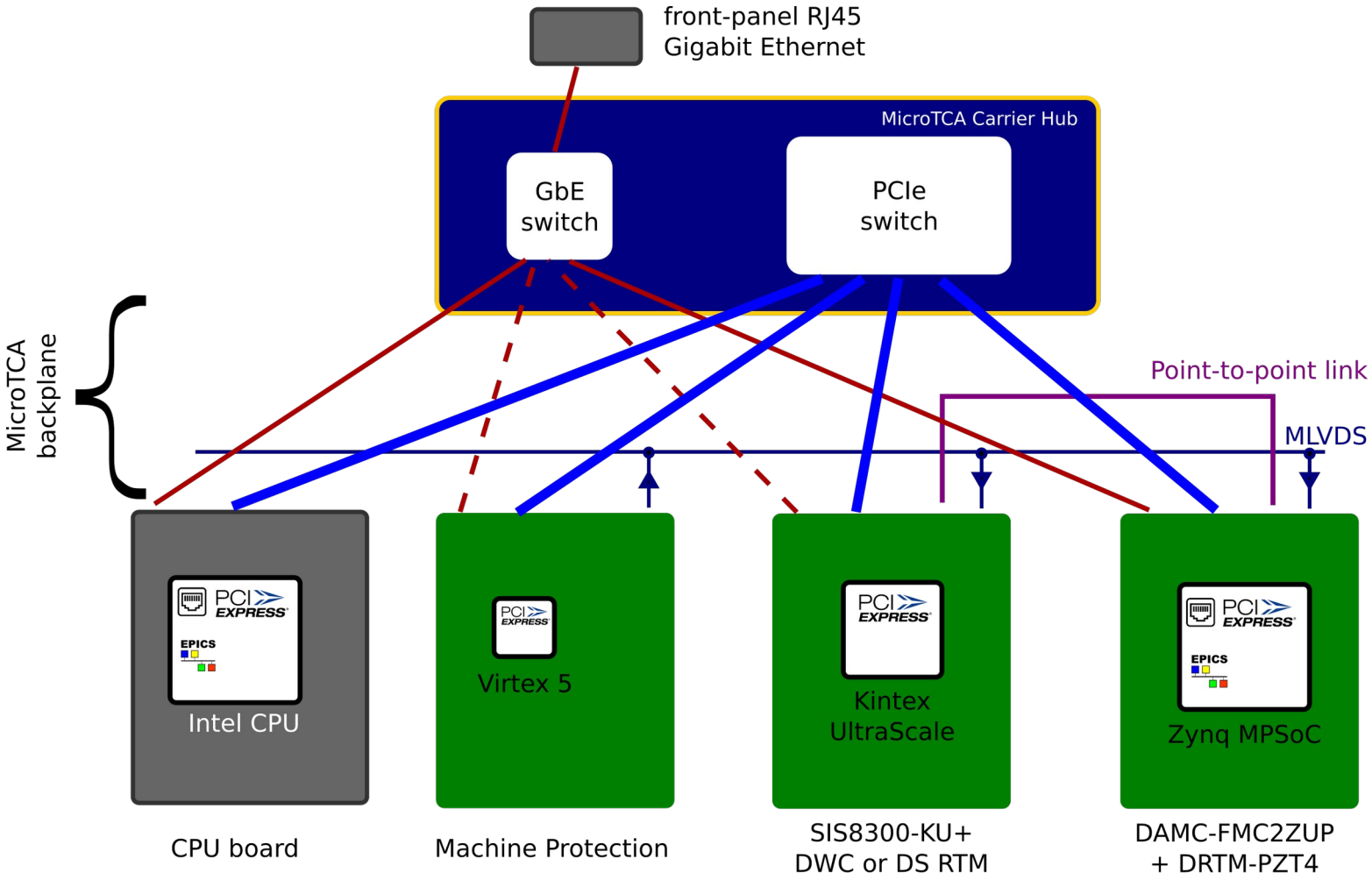

Down Converter (DWC)

Digitizer (ADC)

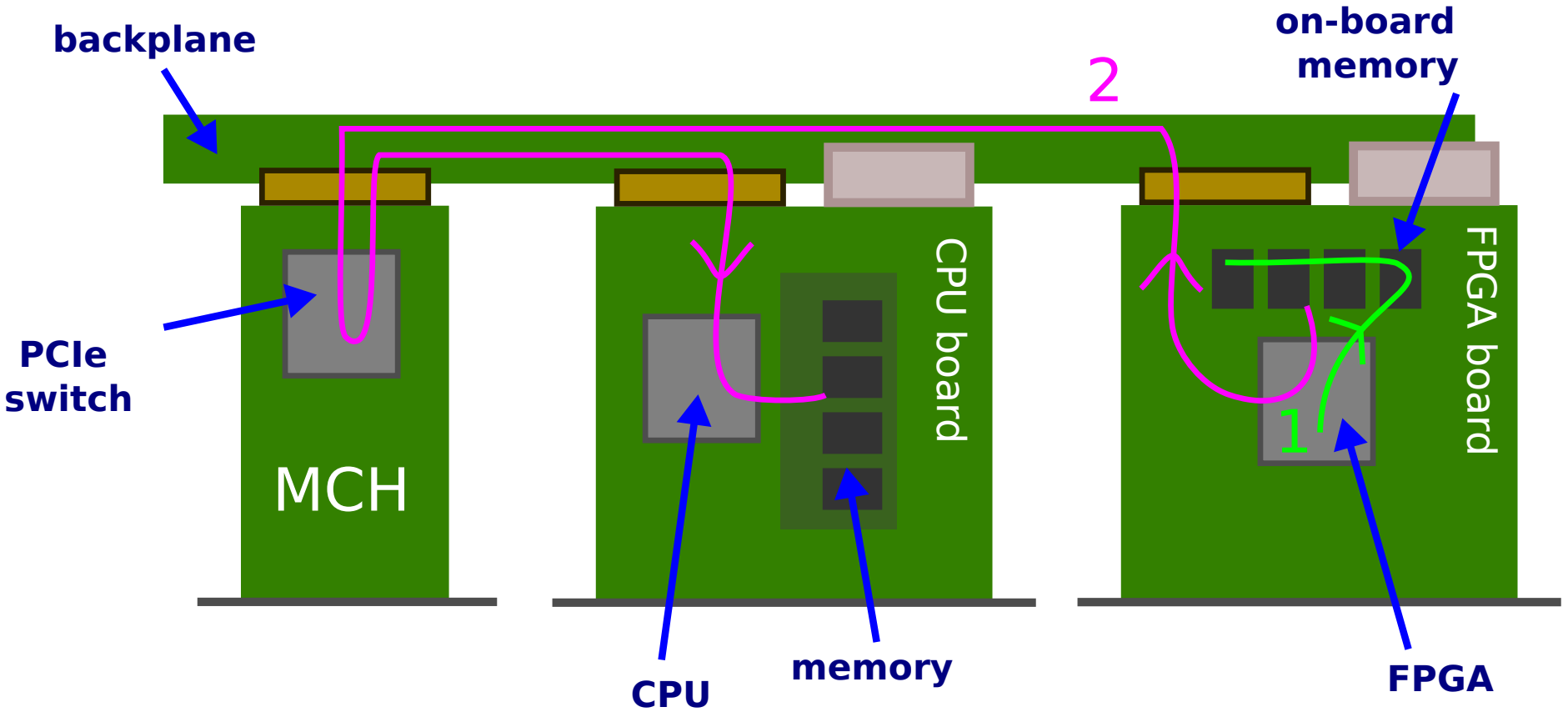
Julien Branlard | LLRF installation for the European XFEL | 7.12.2016 | Page 6

# Visulize how the data moves inside the MTCA Crate

# Document the Data Transfer inside Crate



# Document the Data Transfer inside Crate



# Know your clocking options

# MLVDS

- Multipoint LVDS is used in MicroTCA for communication between cards.
- Ports 17-20 Can be used to forward **clocks, triggers and interlocks** to all other cards on the crate.
- Mesh Topology
  - One AMC acts as a driver
  - other cards can be configured as receivers.
- Wired OR is also possible in MLVDS
  - more than one card can drive the same line (with the same polarity)

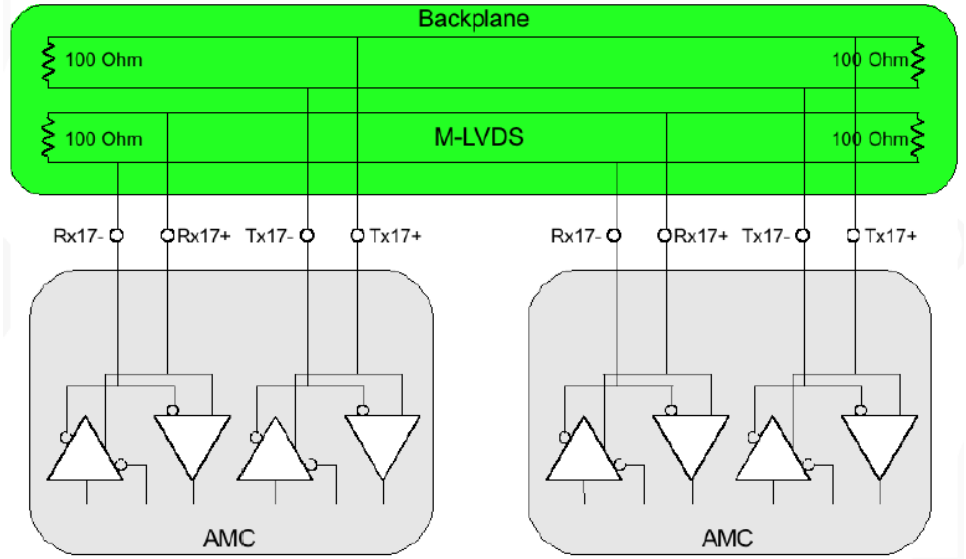


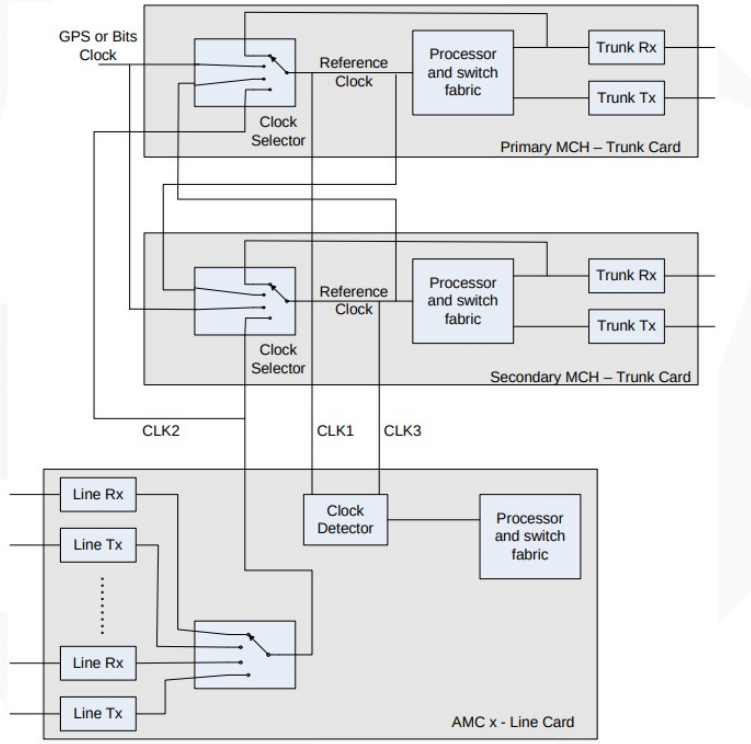

**Figure 6-4: M-LVDS transceiver shown for port 17**

**Table 6-1: Example usage of the 8 bus lines for triggers, interlocks and clocks**

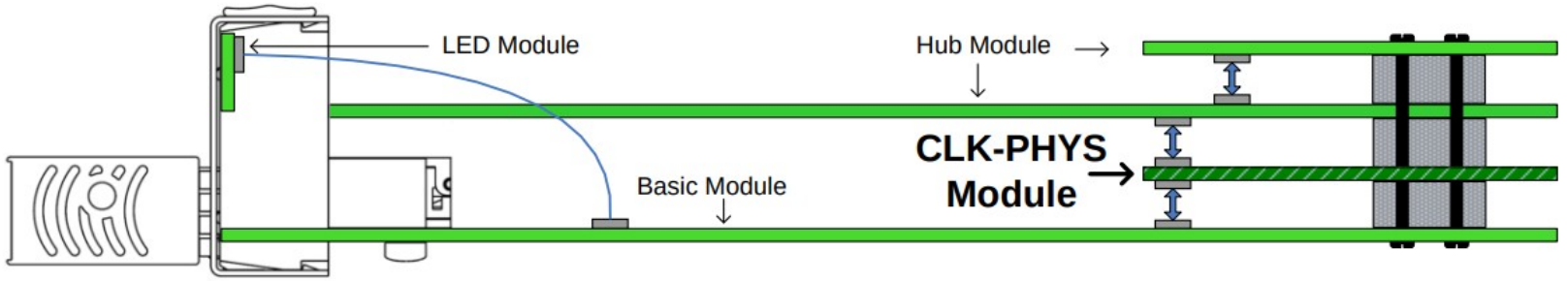
| AMC Port | Name | Description | Usage |
|---|---|---|---|
| Rx17 | TrigStart | Start sampling data | Triggers |
| Tx17 | TrigEnd | Stop sampling data | |
| Rx18 | TrigReadOut | Start data transfer to CPU | |
| Tx18 | ClkAux | Low performance clock | |
| Rx19 | Reset | Reset of counter, dividers | |
| Tx19 | Interlock 0 | Interlock line 0 | 3 interlocks to provide 2 out of 3 redundancy |
| Rx20 | Interlock 1 | Interlock line 1 | |
| Tx20 | Interlock 2 | Interlock line 2 | |

# Clock Distribution inside MicroTCA

- MCH can be used to distribute clocks inside the MicroTCA crate
- TCLKA/B/C/D and FCLK can be generated
- Specially useful for synchronizing multiple AMCs



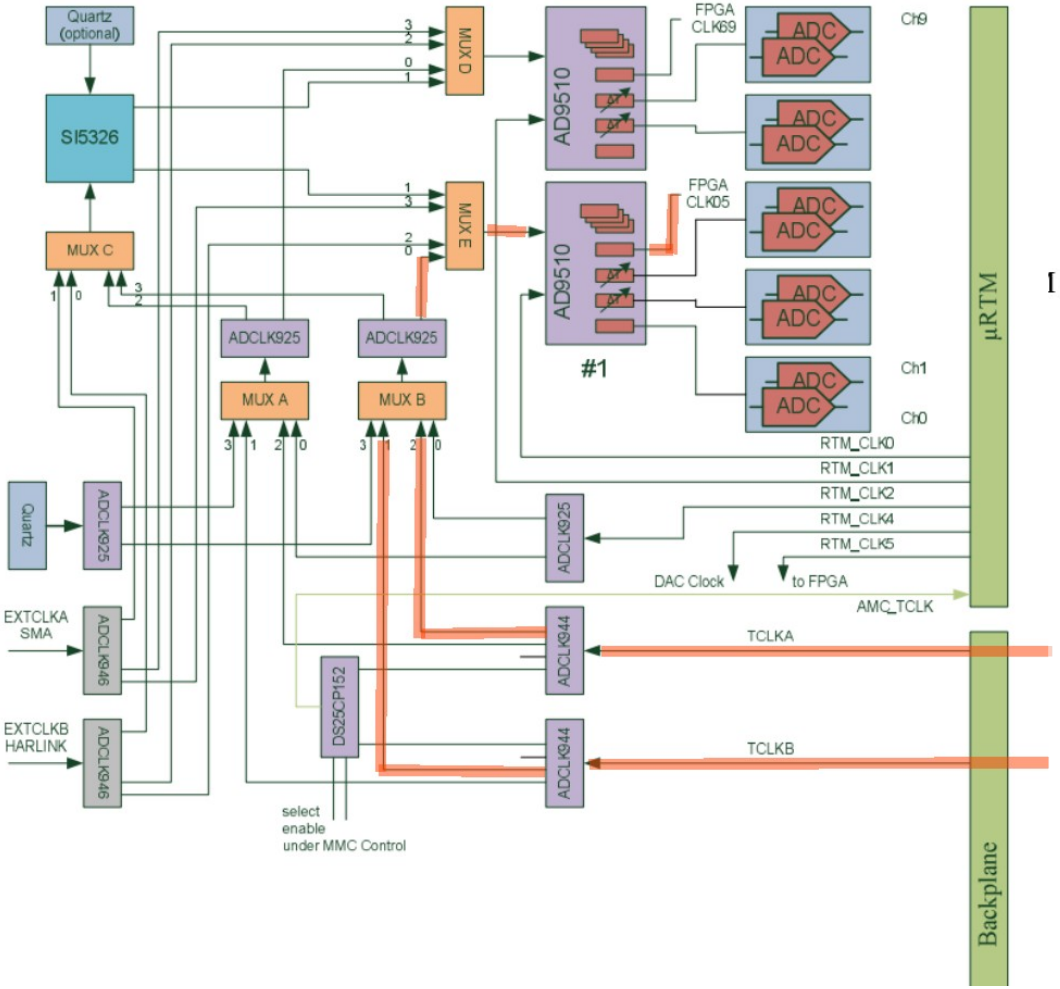

PICMG AMC.0 Specification

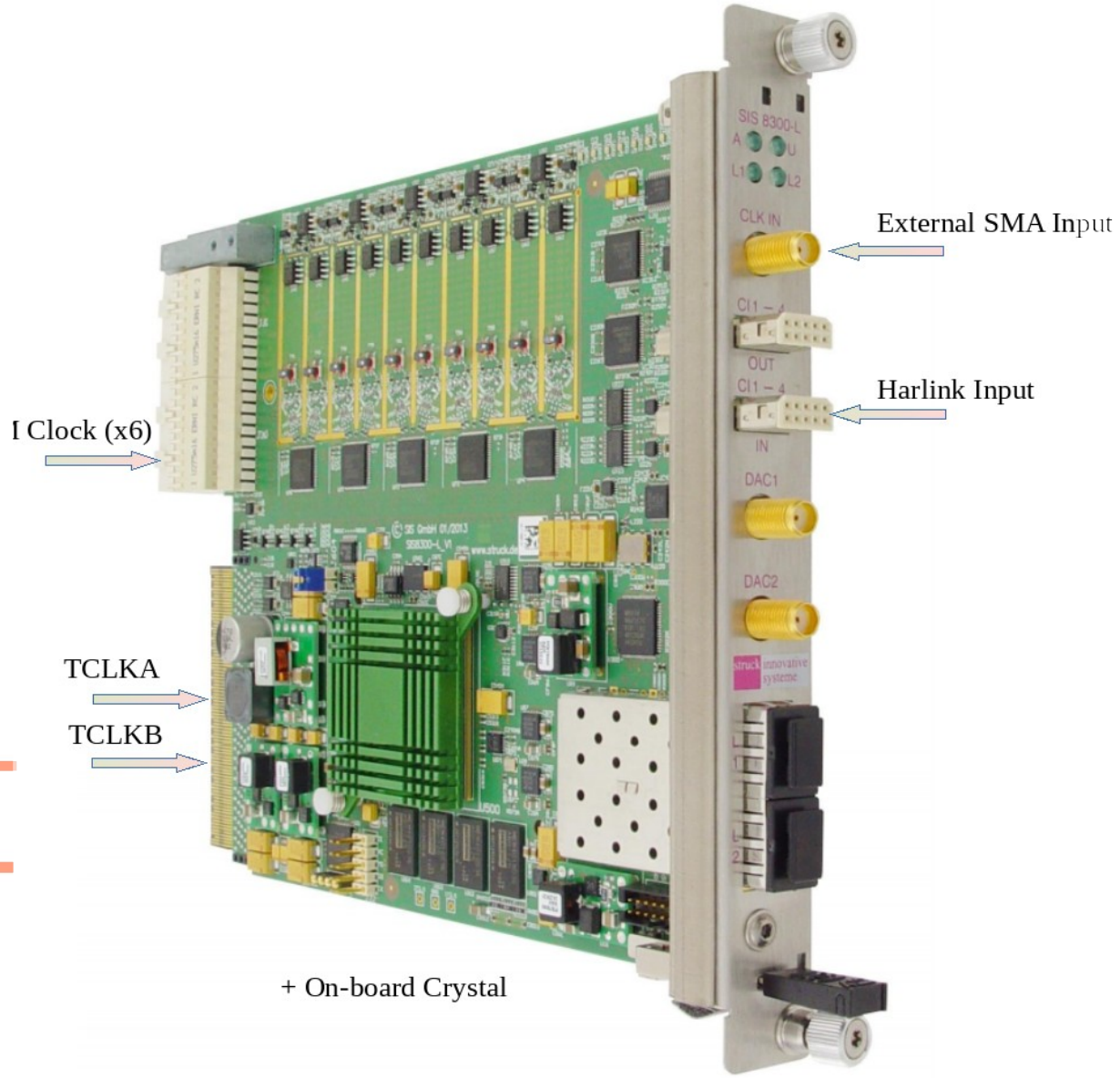



NAT-MCH CLK-PHYS-Module – Technical Reference Manual

# Clocking Options for an AMC

## Case in point: SIS8300-L2 from Struck GmbH

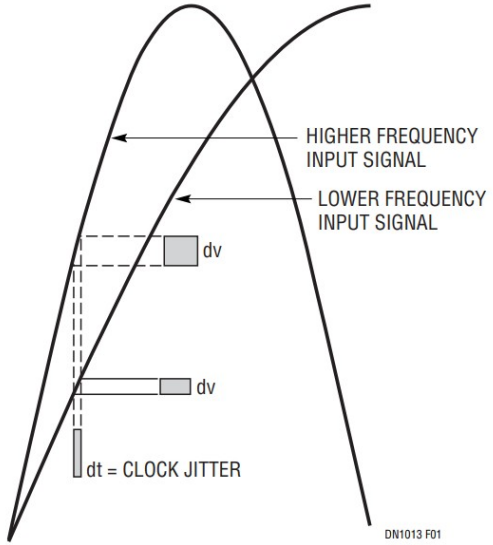


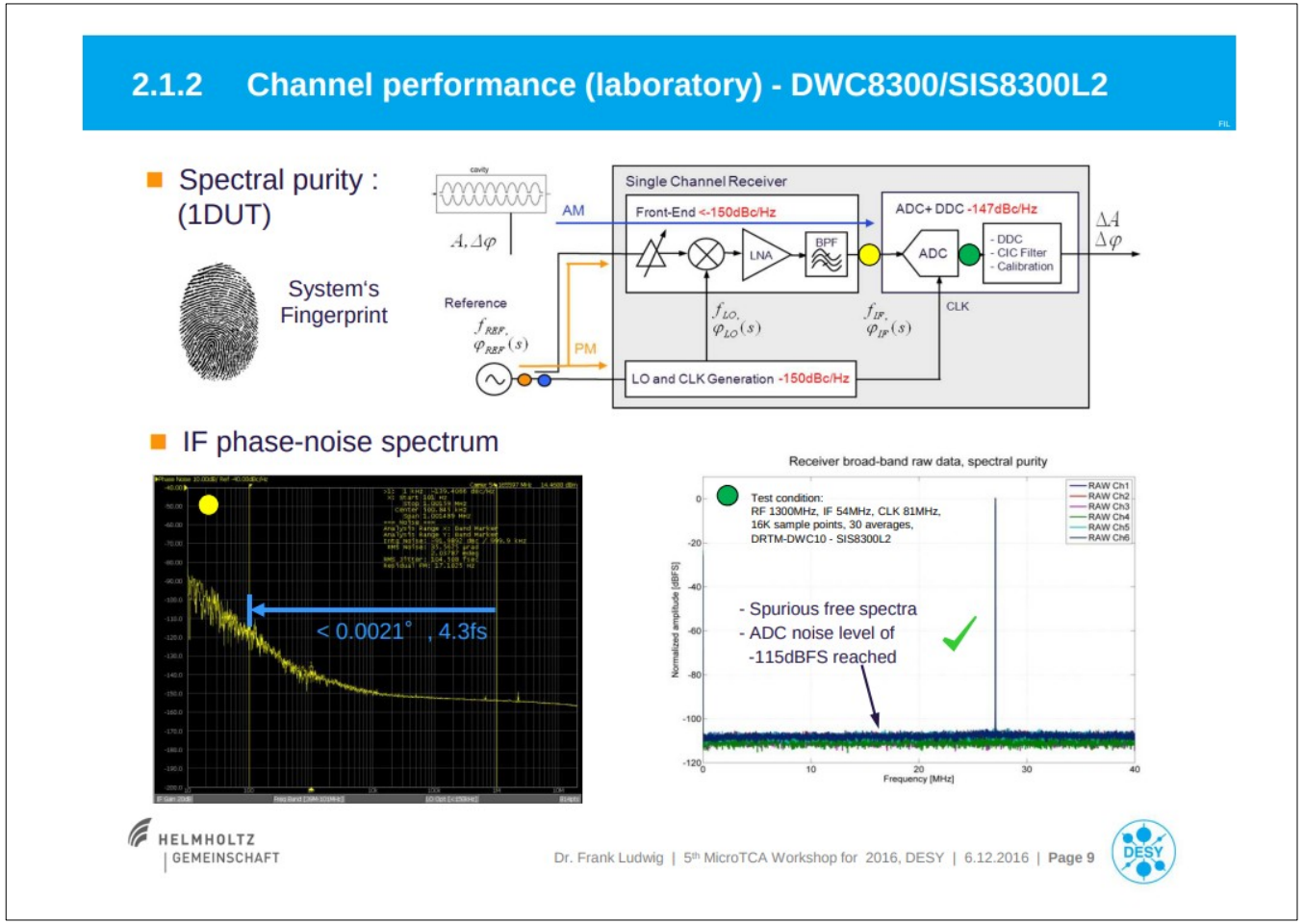
# Clock Jitter effects on ADCs

For digitizers with high input frequencies, jitter of the ADC clock becomes important.

The amount of clock jitter will set the maximum SNR that you can achieve for a given input frequency




Linear Technology | Understanding the Effect of Clock Jitter on High Speed ADCs Design Note 1013

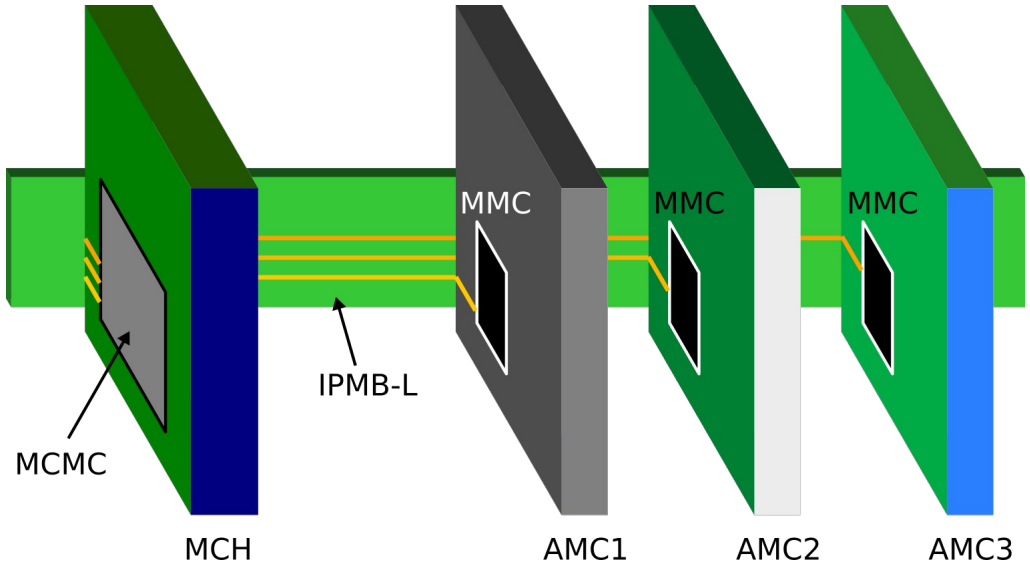


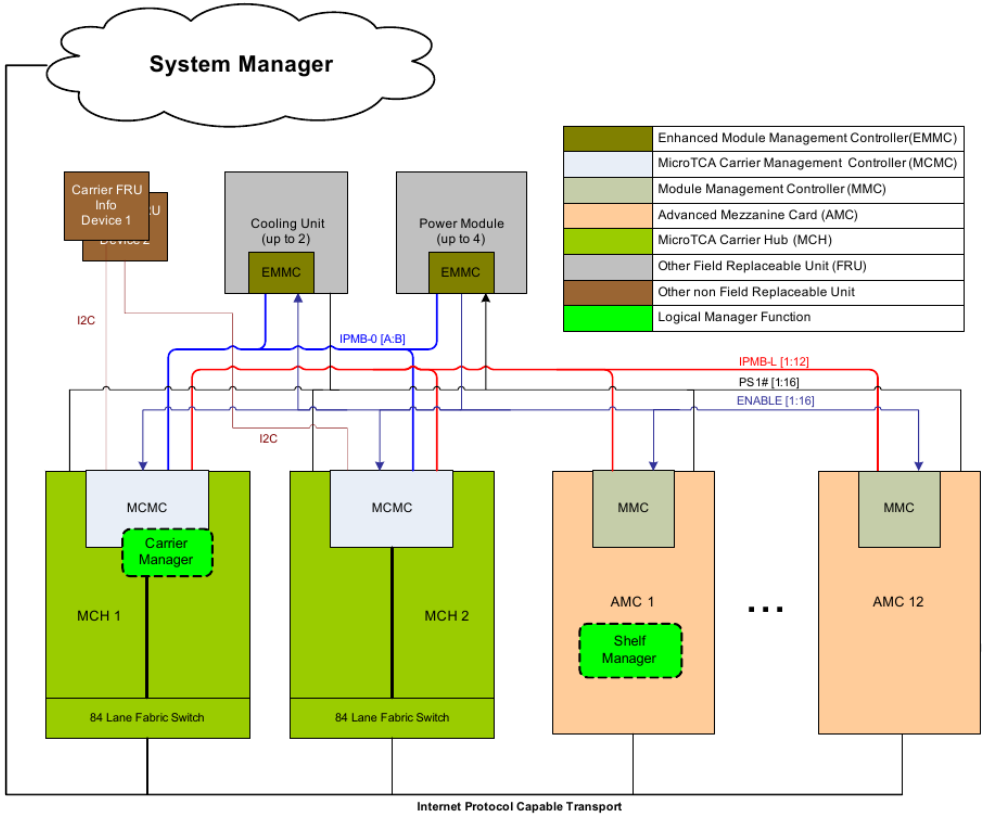

Dr. Frank Ludwig | 5th MicroTCA Workshop | High Performance Measurement Applications in MicroTCA.4

# Learn how to use IPMI

# What is IPMI?

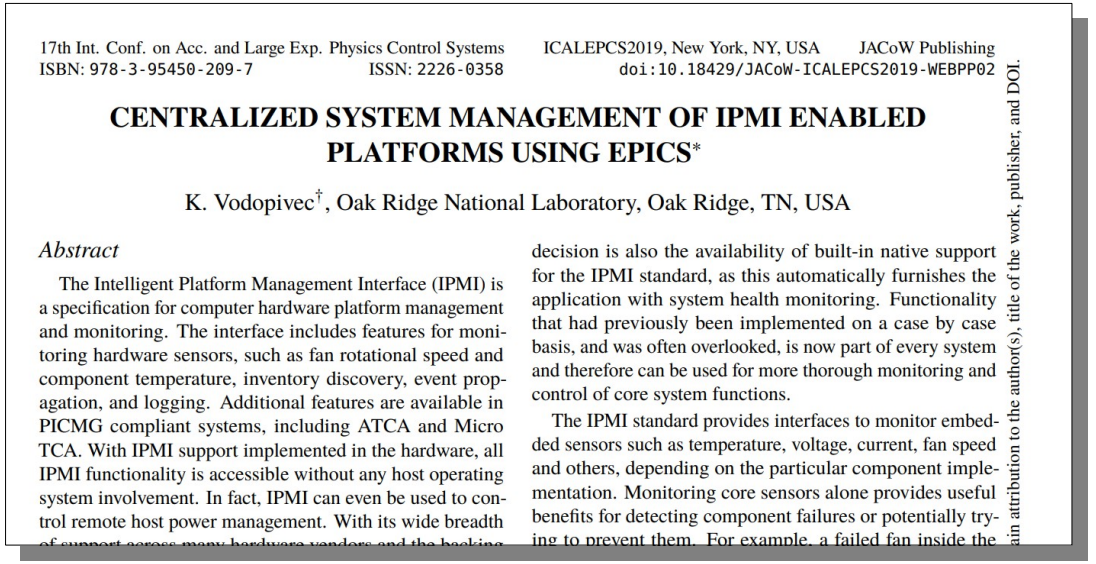
- MicroTCA Standard uses: IPMI (Intelligent Platform Management Interface) for management.
- Specification is led by Intel. Widely used in computer system vendors.
- I2C based protocol that is message-based interface



# Use Open Source tools for IPMI

- 3 main projects for open-source tool (Windows/Linux) for controlling IPMI-enabled systems:
  - ipmitool
  - OpenIPMI
  - FreeIPMI
- By-pass MCH and gain full control of the crate
- Abstraction layer between System Manager and System



17th Int. Conf. on Acc. and Large Exp. Physics Control Systems ICALEPCS2019, New York, NY, USA JACoW Publishing
ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEBPP02

**CENTRALIZED SYSTEM MANAGEMENT OF IPMI ENABLED PLATFORMS USING EPICS***

K. Vodopivec†, Oak Ridge National Laboratory, Oak Ridge, TN, USA

*Abstract*

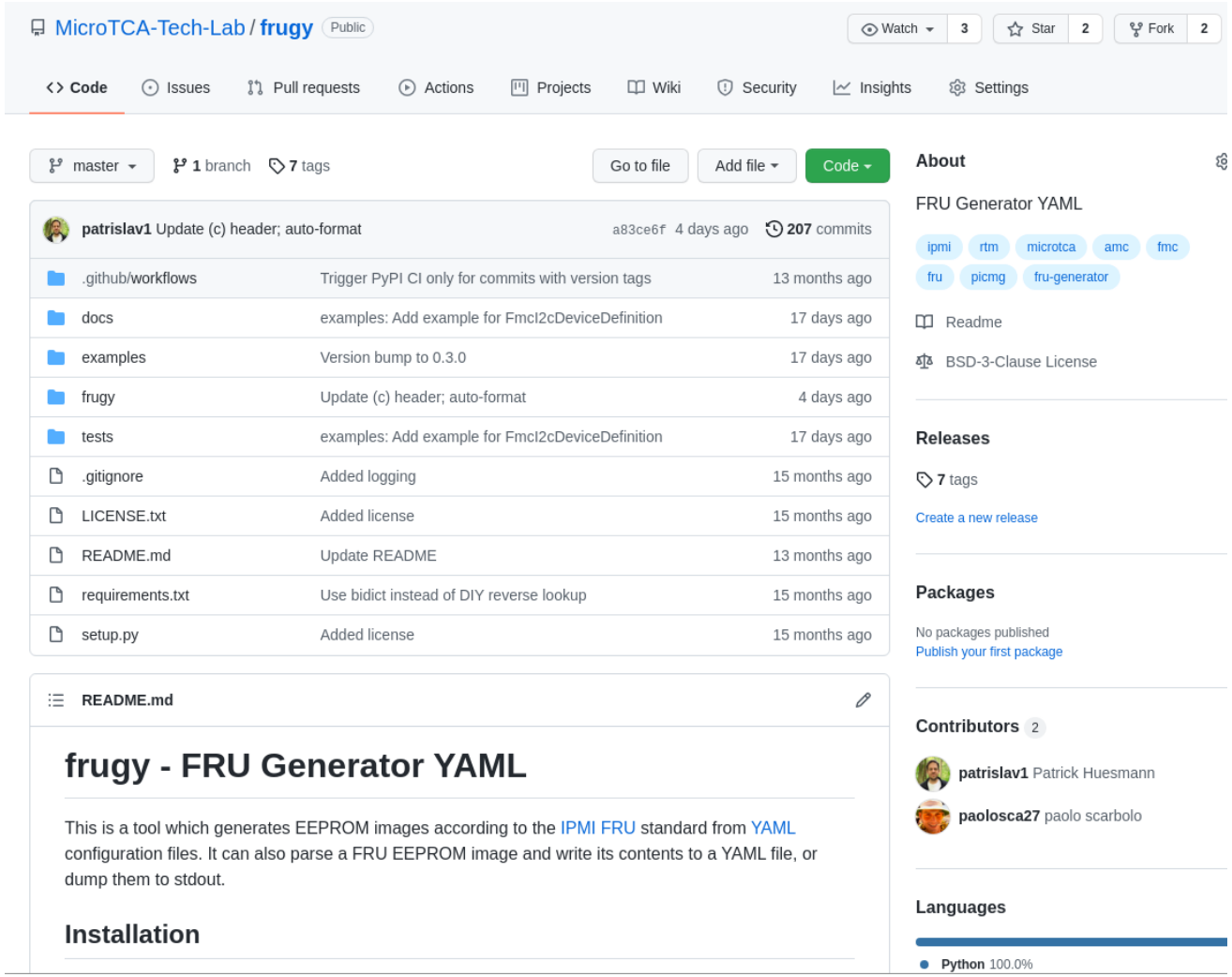
The Intelligent Platform Management Interface (IPMI) is a specification for computer hardware platform management and monitoring. The interface includes features for monitoring hardware sensors, such as fan rotational speed and component temperature, inventory discovery, event propagation, and logging. Additional features are available in PICMG compliant systems, including ATCA and Micro TCA. With IPMI support implemented in the hardware, all IPMI functionality is accessible without any host operating system involvement. In fact, IPMI can even be used to control remote host power management. With its wide breadth of support across many hardware vendors and the backing decision is also the availability of built-in native support for the IPMI standard, as this automatically furnishes the application with system health monitoring. Functionality that had previously been implemented on a case by case basis, and was often overlooked, is now part of every system and therefore can be used for more thorough monitoring and control of core system functions.

The IPMI standard provides interfaces to monitor embedded sensors such as temperature, voltage, current, fan speed and others, depending on the particular component implementation. Monitoring core sensors alone provides useful benefits for detecting component failures or potentially trying to prevent them. For example, a failed fan inside the



# Edit FRU with frugy

- frugy is a **open-souce** tool from MicroTCA-Technology Lab.
- Generated EEPROM images according to the IPMI FRU Standard from **YAML configuration files**.
- Especially useful for people developing a custom AMC board.
- Can be used to ‘edit’ existing FRUs
  - eg. lowering required current for specific AMC on a heavily occupied MTCA crate
  - Edit Inventory information with custom ID for your own company



# IPMI Security

- In today's standards IPMI can be considered 'not secure enough'
- **Several vulnarabilities:**
  - **Insecure input validation**
    - **Bad Privilage Checking**
  - **Shell Injection Vulnerabilities**
  - **Buffer Overflow Vulnerabilities**
- Things to do:
  - Keep IPMI firmware upto date (Even though it is EOL)
  - Change default passwords
  - **NEVER configure IPMI devices on public IP addresses.**
    - Isolate them on a physically separated network.

COMPUTERWORLD UNITED STATES IDG TECH(TALK) COMMUNITY WINDOWS MOBILE OFFICE SOFTWARE APPLE NEWSLETTERS EVENTS

Home > Security

ANALYSIS

**IPMI: The most dangerous protocol you've never heard of**

IPMI could be punching holes in your corporate defenses.

By Paul F. Roberts

ITworld | AUG 19, 2013 7:00 AM PST

You spend thousands or even hundreds of thousands of dollars to secure the data stored on the critical databases and application servers your organization relies on. But what if each of those systems secretly harbored a powerful, hardware based back door that would give a remote attacker total control of the system? And what if that backdoor wasn't planted by some shadowy hacker group operating out of the former Soviet republics, but by the multi-billion dollar Western company that sold you the server in the first place?

**Illuminating the Security Issues Surrounding Lights-Out Server Management**


Anthony J. Bonkoski, *University of Michigan*, *abonkosk@umich.edu*
Russ Bielawski, *University of Michigan*, *jbielaws@umich.edu*
J. Alex Halderman, *University of Michigan*, *jhalderm@umich.edu*



**Abstract**

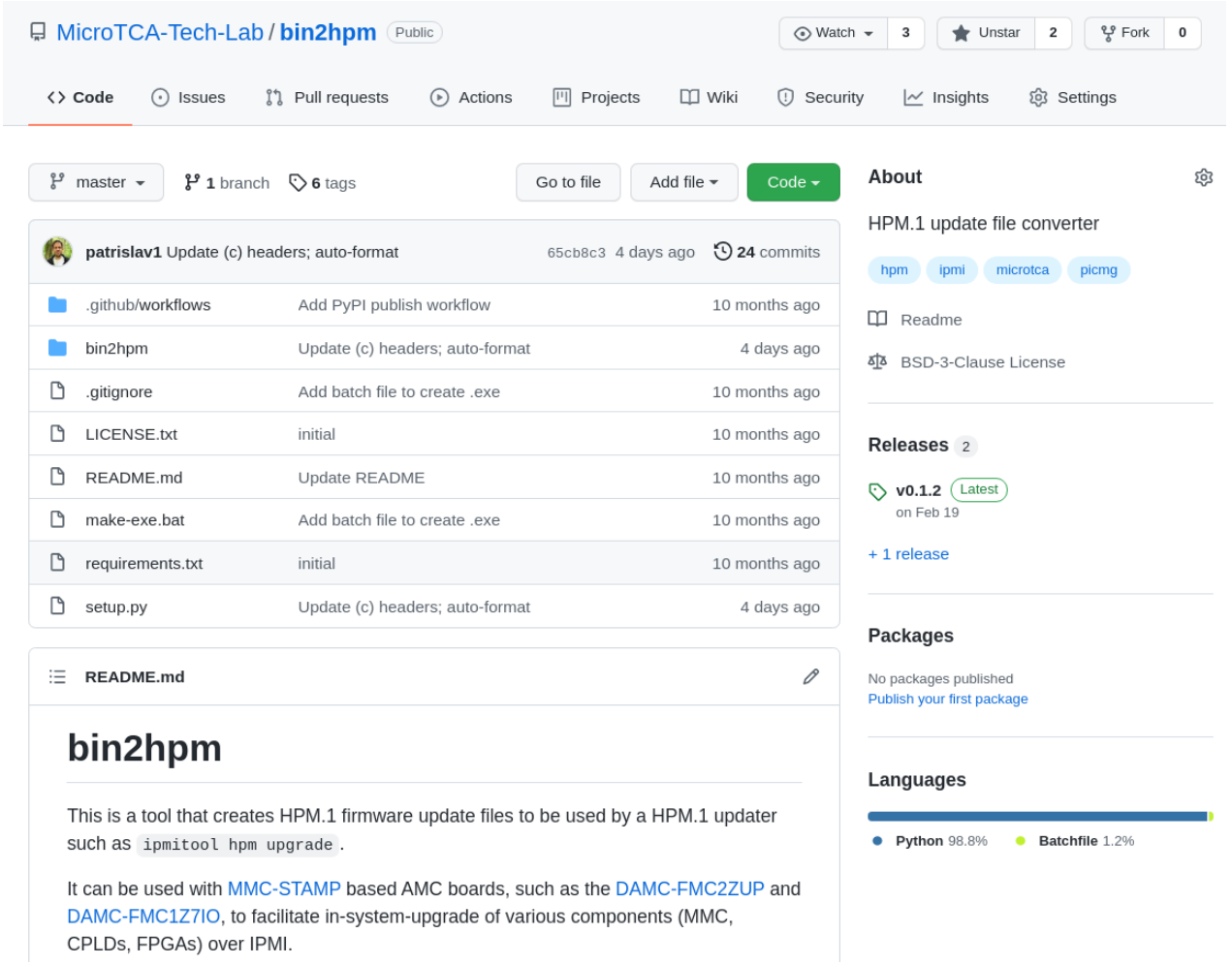
Out-of-band, lights-out management has become a standard feature on many servers, but while this technology can be a boon for system administrators, it also presents a new and interesting vector for attack. This paper examines the security implications of the Intelligent Platform Management Interface (IPMI), which is implemented on server motherboards using an embedded Baseboard Management Controller (BMC). We consider the threats posed by an incorrectly implemented IPMI and present evidence that IPMI vulnerabilities may be widespread. We analyze a major OEM's IPMI implementation and discover that it is riddled with textbook vulnerabilities, some of which would allow a remote attacker to gain root access to the troller that is integrated into the system's motherboard or installed via a daughter card. The BMC has its own flash storage and runs its own operating system, separate from the host's. It typically has access to the PCI bus, to the on-board NIC via a "side-band" interface, and to a collection of sensors and I/O ports [24]. Consistent with its purpose, the BMC has almost total control of the server.

IPMI can be a convenient administrative tool, but, under the control of attackers, it can also serve as a powerful backdoor. Attackers who take control of the BMC can use it to attack the host system and network in a variety of ways. For example, they could install BMC-resident spyware to capture administrative passwords when the operator remotely accesses the host. They could use the

# Exploit all Firmware Upgrade Options

# Firmware Upgrade of AMCs

- Use PCIe/Ethernet to send the bit file to FPGA and trigger reconfiguration. (Xilinx: ICAP)
  - Fast! (~ seconds)
  - If you lose PCIe/Ethernet this method is useless.
- Use HPM (Hardware Platform Management)
  - Created by PICMG
  - Uses IPMI bus to send the firmware data
  - Extremely slow (Ultrascale ~ 1 hr)
  - Can update MMC firmware
- Use JTAG
  - From AMC backplane
  - From JTAG Connector on the PCB



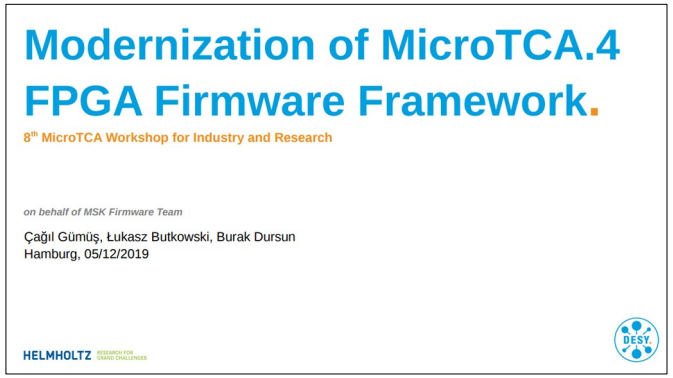
# Use MSK-DESY FPGA Framework:

# FWK

# Update on DESY-MSK FPGA Framework (*FWK*)

## Promises: (Almost) Delivered!

*2 years ago...*




- 8th MicroTCA Workshop: *"We need to modernize our FPGA Firmware Framework"*
  - Version Control switch to git: Done
  - Complete overhaul of tcl framework: Done
  - Get rid of properity buses: 80% Complete
  - IP-Core mentallity: Done
  - Switch to Documentation as Code: Done
  - Promised to open code to public: **January 2022**
- Lessons learned: Licence issues are not fun.
  - Settled on:
    - CERN Open Hardware Licence v.2.0 (weak)
    - Apache 2.0

- Free BSPs: SIS8300-KU, DAMC-TCK7, DAMC-FMC2ZUP, DAMC-Z7IO ….
- Free Libraries: Many I2C, SPI, ADC,DAC components used in MTCA envioreements, useful math functions etc.
- Free Jenkins Pipeline libraries
- ...

Code will be published on:
`gitlab.desy.de/public`

# ChimeraTK + FWK

- Standardized interface definition between FPGA and software: map files
  - Abstraction between register/interrupt addresses and software logic
  - Memory address
  - ReadWrite/ReadOnly
  - Signed/Unsigned
- Automatic generation from the FPGA project by FWK
- Tutorial by Martin Killenberg on ChimeraTK

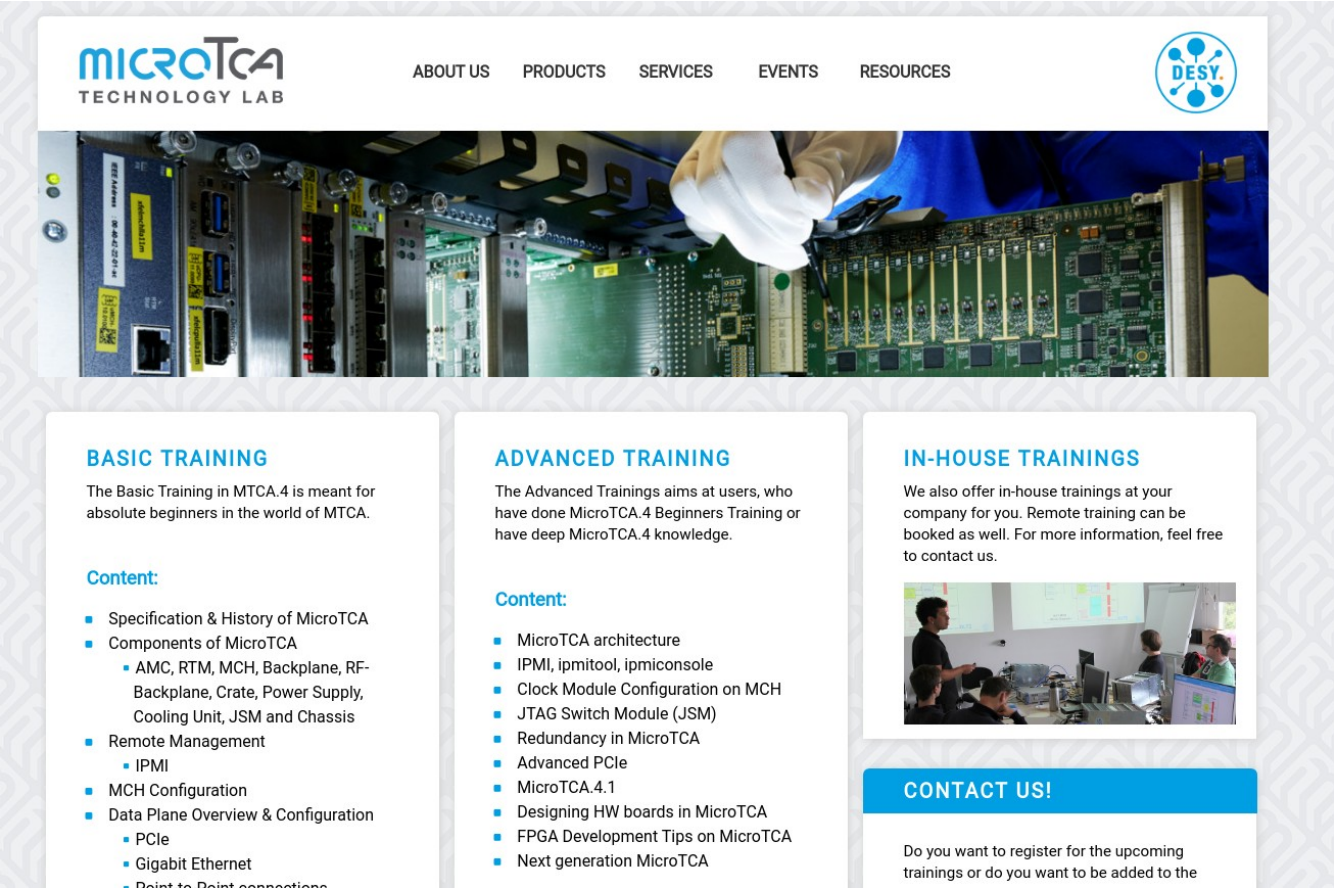
```
@MAPFILE_REVISION 1.0.0-11-g00616703
BOARD.0.WORD_ID                          1      0     4   0  32   0   0  RO
BOARD.0.WORD_VERSION                     1      4     4   0  32   0   0  RO
BOARD.0.WORD_PRJ_ID                      1      8     4   0  32   0   0  RO
BOARD.0.WORD_PRJ_VERSION                 1     12     4   0  32   0   0  RO
BOARD.0.WORD_PRJ_SHASUM                  1     16     4   0  32   0   0  RO
BOARD.0.WORD_PRJ_TIMESTAMP               1     20     4   0  32   0   0  RO
BOARD.0.WORD_CLK_FREQ                    8     32    32   0  32   0   0  RO
BOARD.0.WORD_LLL_STATUS                  6    192    24   0  32   0   0  RO
BOARD.0.WORD_LLL_LOOPBACK                6    224    24   0   3   0   0  RW
BOARD.0.WORD_CLK_MUX                     6    128    24   0   2   0   0  RW
BOARD.0.WORD_CLK_RST                     1    152     4   0   1   0   0  RW
BOARD.0.WORD_CLK_SEL                     1    156     4   0   2   0   0  RW
BOARD.0.WORD_CLK_ERR                     1    168     4   0   1   0   0  RO
BOARD.0.WORD_SPI_DIV_SEL                 1    176     4   0   2   0   0  RW
BOARD.0.WORD_SPI_DIV_BUSY                1    180     4   0   1   0   0  RO
BOARD.0.AREA_SPI_DIV                   128   4096   512   0   8   0   0  RW
BOARD.0.AREA_PLL                       512  16384  2048   0   8   0   0  RW
BOARD.0.AREA_PLL_CONF                  512  12288  2048   0  24   0   0  RW
BOARD.0.WORD_PLL_CONF_STR                1    272     4   0   2   0   0  RW
BOARD.0.WORD_PLL_CONF_RST                1    276     4   0   1   0   0  RW
BOARD.0.WORD_ADC_ENA                     1    256     4   0   1   0   0  RW
BOARD.0.WORD_ADC_OV                      1    260     4   0  10   0   0  RO
BOARD.0.WORD_SPI_ADC_SEL                 1    288     4   0   3   0   0  RW
BOARD.0.WORD_SPI_ADC_BUSY                1    292     4   0   1   0   0  RO
BOARD.0.AREA_SPI_ADC                   256   8192  1024   0   8   0   0  RW
BOARD.0.WORD_ADC_FIFO_RESET              1    264     4   0  10   0   0  RW
BOARD.0.WORD_ADC_FIFO_DELAY             10    320    40   0   8   0   0  RW
BOARD.0.WORD_BOOT_STATUS                 1    384     4   0   1   0   0  RW
BOARD.0.AREA_BOOT                    16384  65536 65536   0  32   0   0  RW
BOARD.0.WORD_IDELAY_SEL                  1    464     4   0   8   0   0  RW
BOARD.0.WORD_IDELAY_INC                  1    468     4   0   1   0   0  RW
BOARD.0.WORD_RESET_N                     1    512     4   0   1   0   0  RW
BOARD.0.WORD_USER                        1    516     4   0  32   0   0  RW
BOARD.0.WORD_DAC_ENA                     1    520     4   0   1   0   0  RW
BOARD.0.WORD_CLK_PHASE_INCDEC            1    536     4   0   1   0   0  RW
BOARD.0.WORD_DAC_IDELAY_SEL              1    540     4   0   1   0   0  RW
BOARD.0.WORD_DAC_IDELAY_INC              1    544     4   0   1   0   0  RW
BOARD.0.WORD_DAC_IDELAY_CNT              1    548     4   0   5   0   0  RO
BOARD.0.WORD_ADC_IDELAY_CNT              5    592    20   0   5   0   0  RO
```

| | | | |
|---|---|---|---|
| **Break** | | | |
| *Virtual Workshop via Zoom* | | | 13:10 - 13:20 |
| **Developing for SoC-based AMCs** | *Jan Marjanovic* | **ChimeraTK** | *Martin Killenberg* |
| *Virtual Workshop via Zoom* | 13:20 - 13:55 | | |
| **PCIe and Open Source Linux Driver** | *Ludwig Petrosyan* | | |
| *Virtual Workshop via Zoom* | 13:55 - 14:30 | *Virtual Workshop via Zoom* | 13:20 - 14:30 |
| **Break Breakfast / Lunch / Dinner** | | | |

# Do you want to learn more?

## Let us show you how deep the rabbit hole goes

- Go to techlab.desy.de to learn more about the training
- 2 Trainings:
  - Basic
  - Advanced
- Dates for 2022 will be announced soon!
- Training can be held virtually/in-house depending on the health guidance.



# Thank you

## Contact

DESY. Deutsches
Elektronen-Synchrotron

www.desy.de

Çağıl Gümüş (CJ)
FPGA Team MSK DESY
cagil.gumues@desy.de
+49408998 3760