

# MTCA4U — The DESY MicroTCA.4 User Tool Kit.

Generic software and drivers for MicroTCA.4 based controls

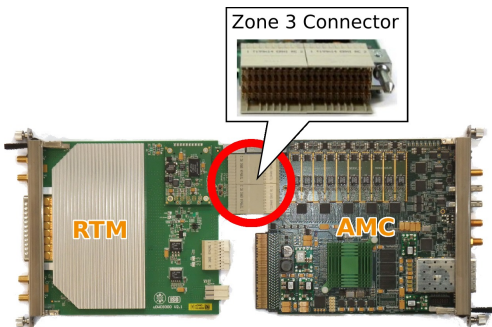
**Martin Killenberg**



L. Petrosyan, C. Schmidt, *DESY, Hamburg, Germany*  
S. Marsching, *aquenos GmbH, Baden-Baden, Germany*  
M. Mehle, T. Sušnik, K. Žagar, *Cosylab d.d., Ljubljana, Slovenia*  
A. Piotrowski, *FastLogic Sp. z o.o., Łódź, Poland*  
T. Kozak, P. Prędkie, J. Wychowaniak, *Łódź University of Technology,  
Łódź, Poland*

## Based on Advanced Telecommunications Computing Architecture (ATCA)

- High speed serial bus
- High modularity due to Advanced Mezzanine Cards (AMCs)
- Redundancy
- Hot-swap capability



## MicroTCA.4 Enhancements

Application in physics and accelerator control

- Rear Transition Modules (RTMs)
- AMC-RTM connection via *Zone 3*
- Allows separation of analog and digital processing
- Clock and trigger lines
- Low latency point to point serial I/O
- Advanced shelf management

ATCA and MicroTCA are industry standards defined by the PCI Industrial Computer Manufacturers Group (PICMG).

## Goal

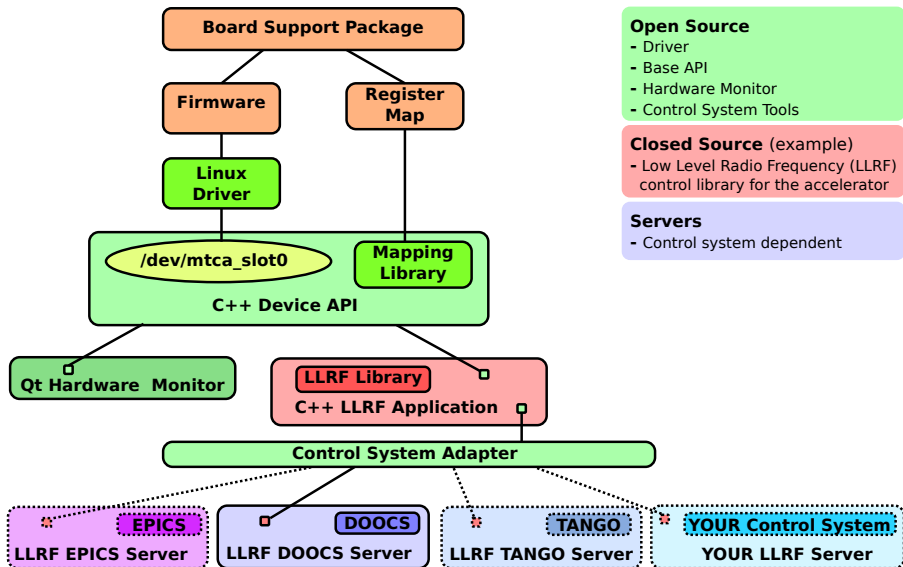
Provide a tool kit to facilitate the development for MicroTCA.4 based control applications.

MTCA4U comprises

- Linux drivers for PCIeexpress
- Intuitive C++ API
- Tools for easy integration into control systems
- Board-specific classes for implementations used at DESY

## Requirements

- Independent from the control system
- Universal and extensible
- Base version open source (compile on many distributions)
- Board-specific classes can be closed source (protection of intellectual property)



## Open Source

- Driver
- Base API
- Hardware Monitor
- Control System Tools

## Closed Source (example)

- Low Level Radio Frequency (LLRF) control library for the accelerator

## Servers

- Control system dependent

## Required functionality is the same for all boards

- Read and write access to registers on the FPGA
- Direct Memory Access for large data transfers (needs firmware support)
- Atomic read–modify–write of registers via ioctl

## Generic base driver

- Implements common functionality
- Functions exported to kernel space
- High re-usability of code
- Only board-specific ioctls have to be added

## Universal driver

- Standard register set for all boards
- DMA support for DESY firmware

## PICMG Software Working Group

Recommendation for a Standard Hardware API which defines a common register set for all boards

## Basic C++ API

- Classes for convenient read/write via PCIeexpress
- Interface for Direct Memory Access (no need to bother with driver implementation details)
- Register name mapping

## Register Name Mapping

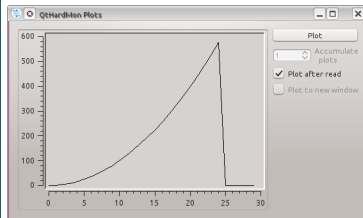
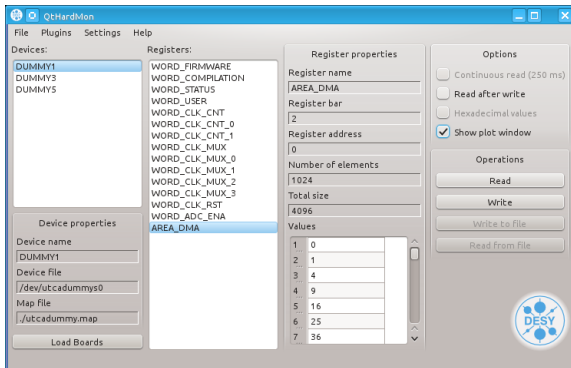
Mapping file for the specific firmware

- Automatically generated by the firmware board support package
- Contains information about
  - Register name
  - Address
  - Size
  - Data type

### Advantages:

- Use descriptive names instead of hex-addresses
- Better code readability
- User code becomes independent from firmware version
- Automated type conversion

## GUI for the basic API



- Display devices and registers by name
- Show and modify register content
- Basic plotting functionality

## Task

Complex control algorithms should be used with different control systems.

## Contradicting requirements

- Keep application code control system independent
  - The algorithm must interact with the control system
  - Do not re-implement functionality provided by the control system
- ⇒ Keep the layer as thin as possible

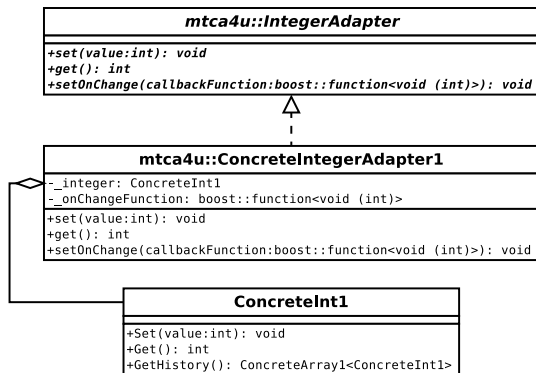
## Additional requirements:

- Thread-safety
- Real-time capability
- Must not copy large data objects (arrays)

## Control System Adapter

- Process variables to transfer data to/from the control system
- Callback mechanism to perform actions





- Wrapper classes for
  - simple data types
  - arrays of simple data types
- Contains instance of control system variable
- Only basic interface
  - Get function
  - Set function
  - Callback function on change
  - **No control system functionality!**
  - Random access operator and iterators for arrays

## Three types of actions

- ① Synchronous actions on set/get
    - Only one process variable is updated
    - Callback function registered with each process variable
  - ② Update functions triggered by the control system
    - Periodic updates
    - Control system triggers
    - All process variables are updated
    - Callback function registered with the control system adapter
  - ③ Synchronisation triggered by the business logic
    - Control loop running in its own thread
    - Business logic determines when to synchronise
    - All process variables are updated
    - Callback function registered with the control system adapter
- The control system adapter assures thread safety for all process variables inside the callback functions.
  - All callback functions have a control system independent signature.

## Engineering versions for

- Board support package and firmware
- Universal PCIexpress driver
- C++ I/O class
- Register name mapping
- Hardware monitor GUI

## Under development

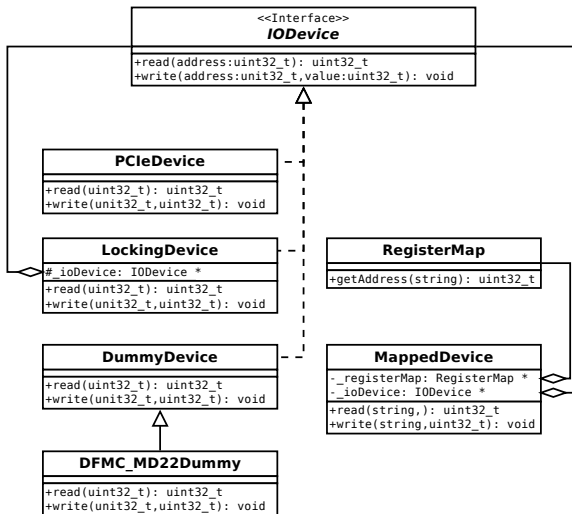
- Control system adapter
- Python bindings
- Quality control (fully automated unit tests)

MTCA4U is published under the GNU General Public License.

Subversion repository on the DESY svn server:

<https://svnsrv.desy.de/public/mtca4u/>

# Backup



## Modern, object oriented design

- Easy to use interfaces
- Multiple abstraction layers, adapted to the different use cases
  - Normal operation
  - Calibration/setup
  - Expert

## Unit testing framework

- Well tested code
- Facilitates refactoring
- Dummy devices for software development without hardware access
- Code coverage

## Doxygen documentation

- Complete, browsable API documentation

DFMC-MD22 Motor Controller: mtca4u::MotorDriverCard Class Reference – Konqueror

File Edit View Go Bookmarks Settings Window Help

/space/killenb/MotorDriverCard\_trunk/doc/html/classmtca4u\_1\_1\_motor\_driver\_card.html

mtca4u > MotorDriverCard >

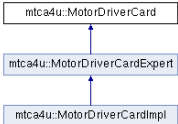
Public Member Functions

## mtca4u::MotorDriverCard Class Reference

A class to access the DFMC-MD22 motor driver card, which provides two MotorControllers. More...

```
#include <MotorDriverCard.h>
```

Inheritance diagram for mtca4u::MotorDriverCard:



```

classDiagram
    class mtca4u_MotorDriverCardExpert["mtca4u::MotorDriverCardExpert"]
    class mtca4u_MotorDriverCardImpl["mtca4u::MotorDriverCardImpl"]
    class mtca4u_MotorDriverCard["mtca4u::MotorDriverCard"]
    mtca4u_MotorDriverCardExpert --|> mtca4u_MotorDriverCard
    mtca4u_MotorDriverCardImpl --|> mtca4u_MotorDriverCardExpert
  
```

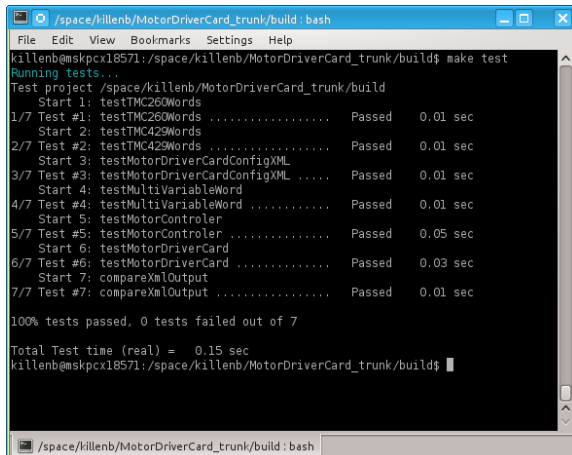
List of all members.

### Public Member Functions

virtual unsigned int	<b>getControlerChipVersion</b> ()=0
virtual <b>MotorControler</b> &	<b>getMotorControler</b> (unsigned int motorControlerID)=0 Get acces to one of the two motor controlers on this board.
virtual <b>PowerMonitor</b> &	<b>getPowerMonitor</b> ()=0 Get a reference to the power monitor.
virtual <b>ReferenceSwitchData</b>	<b>getReferenceSwitchData</b> ()=0

### Detailed Description

- Tests written using the boost::test library
- Fully integrated into the CMake build system
  - Automatically run when packaging, e.g.
- Used to create code coverage report
  - Goal: Test every single line of code



```
killenb@mskpcx18571:/space/killenb/MotorDriverCard_trunk/build$ make test
Running tests...
Test project /space/killenb/MotorDriverCard_trunk/build
  Start 1: testTMC260Words
1/7 Test #1: testTMC260Words ..... Passed    0.01 sec
  Start 2: testTMC429Words
2/7 Test #2: testTMC429Words ..... Passed    0.01 sec
  Start 3: testMotorDriverCardConfigXML
3/7 Test #3: testMotorDriverCardConfigXML ..... Passed    0.01 sec
  Start 4: testMultiVariableWord
4/7 Test #4: testMultiVariableWord ..... Passed    0.01 sec
  Start 5: testMotorController
5/7 Test #5: testMotorController ..... Passed    0.05 sec
  Start 6: testMotorDriverCard
6/7 Test #6: testMotorDriverCard ..... Passed    0.03 sec
  Start 7: compareXmlOutput
7/7 Test #7: compareXmlOutput ..... Passed    0.01 sec

100% tests passed, 0 tests failed out of 7

Total Test time (real) =  0.15 sec
killenb@mskpcx18571:/space/killenb/MotorDriverCard_trunk/build$
```

