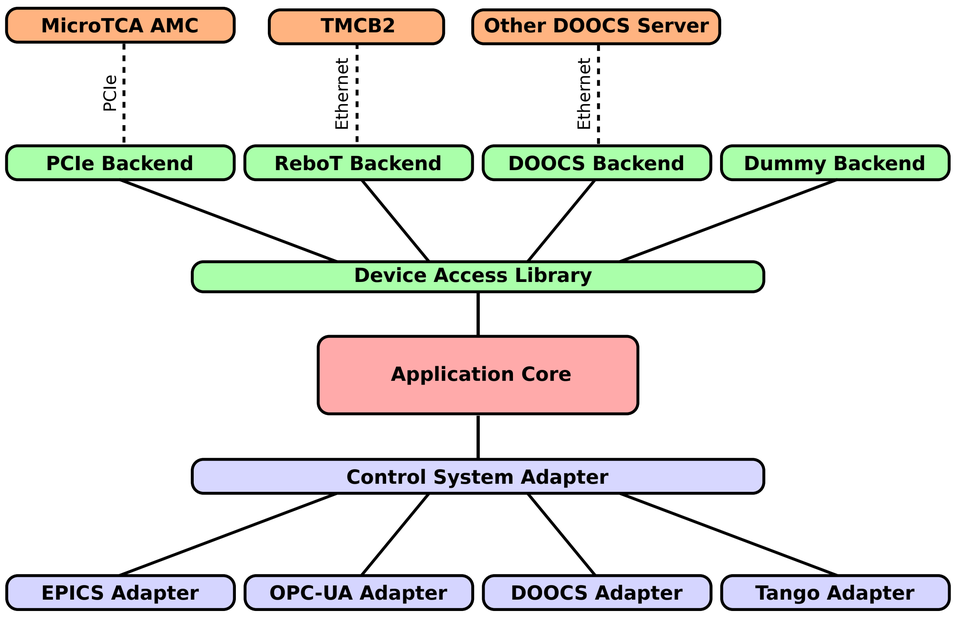
ChimeraTK: A toolkit for modular control applications.

M. Killenberg, M. Heuer, M. Hierholzer, L. Petrosyan, C. Schmidt, N. Shehzad, T. Kozak, G. Varghese, M. Viti (DESY, Germany), S. Marsching (aquenos GmbH, Germany), A. Piotrowski (FastLogic, Poland), P. Prędki, J. Wychowaniak (Łódź University of Technology, Poland), A. Dworzanski, K. Czuba (Warsaw University of Technology, Poland)

Overview

- Low-level RF controls and special diagonstics for XFEL and FLASH need a large number of complex devices
- Heterogenous architecture: FPGAs, microcontroller, frontend and middelware PCs connected via PCIe, Ethernet, etc.
- Devices of same type are used in different environments and setups even within the same accelerator
- Technology should also be used in other facilities (ELBE at HZDR, FLUTE at KIT etc.)
- Framework is needed to abstract applications from the exact setup and even the facility
- Using modern C++ 11
- Everything is open source, (L)GPL license



MTCA4U → ChimeraTK

ChimeraTK was formerly known as the MicroTCA.4 user tool kit.

Presenter

Martin Hierholzer
DESY
22607 Hamburg

martin.hierholzer@desy.de

Device Access

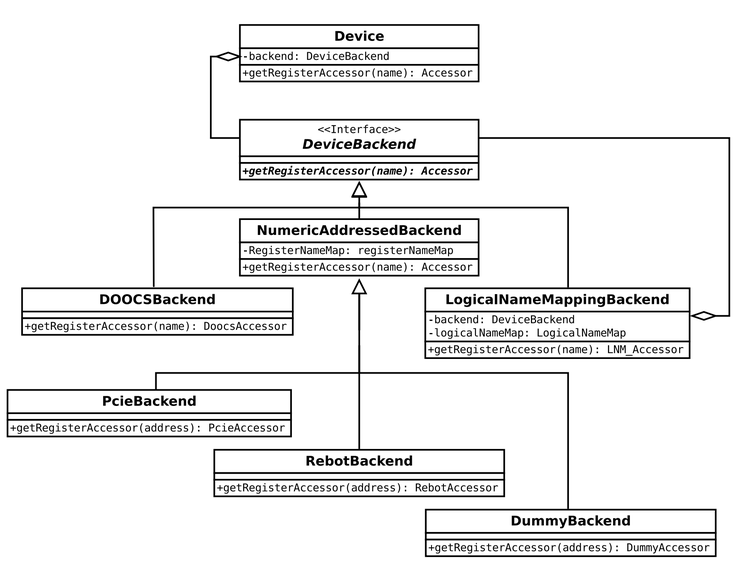
- Library for accessing any register-based device
- Abstract backend interface: access devices through different transport layers with a single interface
- Runtime register name mapping: Access registers by name even if target uses numeric addresses (map file e.g. from board support package)
- Runtime device name mapping: Devices are mapped to alias names (independent of backend type)
- Transparent data conversion

Available backends

- PCI-Express (e.g. for MicroTCA.4 devices)
- Simple network protocol for FPGAs: RebоT (Register-based over TCP)
- DOOCS client
- DummyBackend / VirtualLab (see talk WECSPLCO03)
- Logical name mapping for more abstraction from implementation details of the firmware

Tell us what you need!

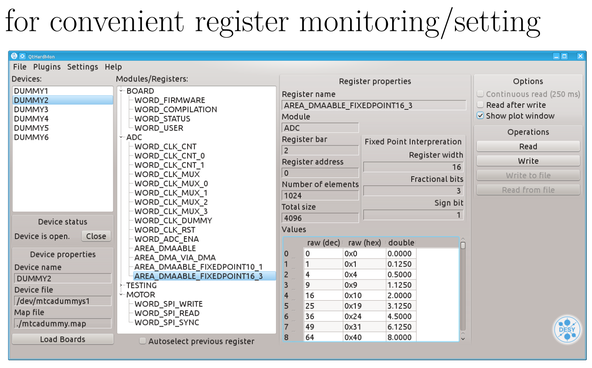
Class structure



- Actual access to data through "Accessors" with a variable- or vector-like interface

Tools and language bindings

- Language bindings to Matlab and Python
- Command line tools for scripting
- GUI for convenient register monitoring/setting



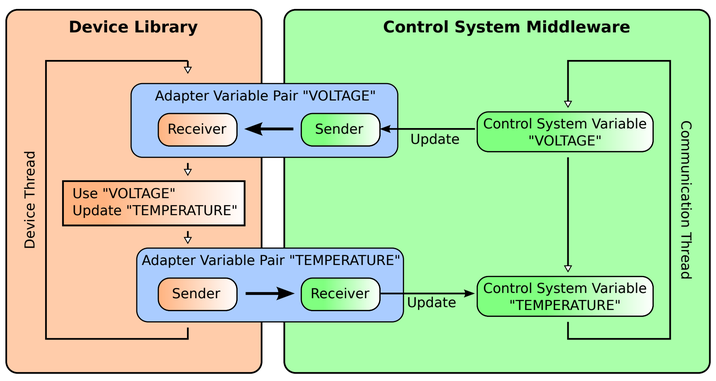
Status

- Well tested and in use for a long time
- Available on Github:
  `http://github.com/ChimeraTK/DeviceAccess`

Control System Adapter

- Library to write control system servers for different middlewares
- Application code fully middleware independent
- Thead-safe, real-time capable, efficient
- Supported middlewares: DOOCS, EPICS 3, OPC-UA
- Extensible to new middlewares

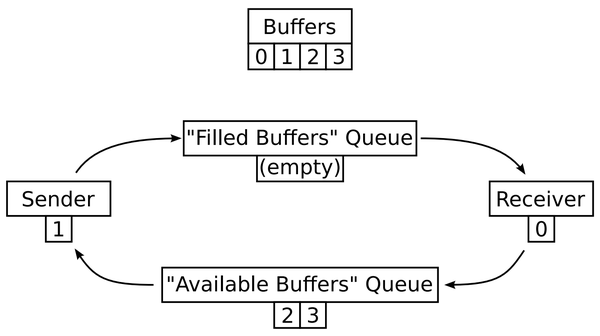
Decoupling of device thread



- Device thread:
  - Device logic independent of middleware
- Middleware thread:
  - Synchronises adapter variables and control system variables
  - Independent of device / application

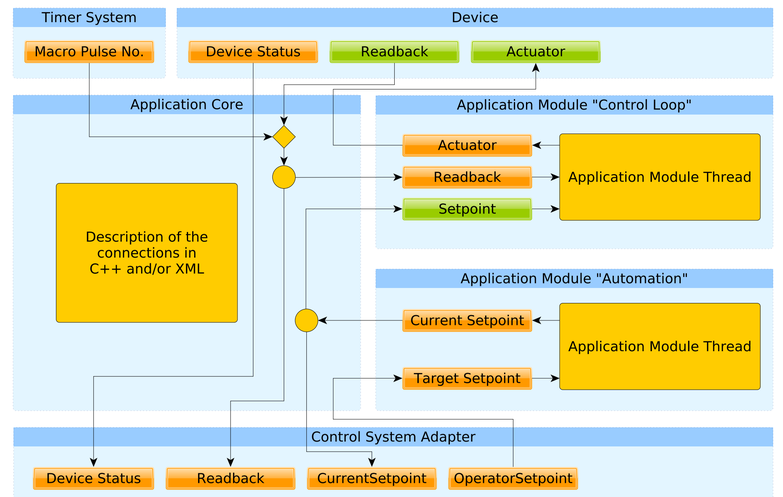
Process variable implementation

- Process variables are sender-receiver pairs
- Each one side belongs to either of the device and middleware threads
- Lock free queues and pre-allocated buffers for real-time capability
- Copy references, not buffers for efficiency



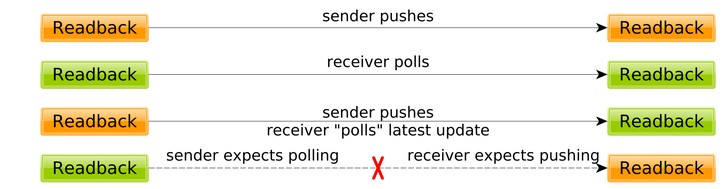
Application Core

- Unified handling of device registers and control system variables
- Decoupling of variable instantiation and algorithms (through *inversion of control*)
- Allows quick structural changes (e.g. move a module into a separate server)
- Efficient coding: avoid boiler-plate code as much as possible

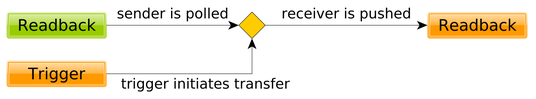


Handling of process variables

- Two types of variables:
  - active sender pushes updates to passive receiver
  - passive sender is polled by active receiver



- A polled sender cannot directly be connected to a pushed receiver
- By using any pushing sender as trigger, the transfer can be initiated



- "Fan out" to distribute variables

Status of the Control System Adapter and the Application Core

- Basic functionality of the Control System Adapter implemented and tested
- Currently under rework for a coherent interface with DeviceAccess ⇒ requirement for ApplicationCore
- ApplicationCore has a proof-of-concept implementation for scalars only
- Extension of ApplicationCore to arrays and with more efficient implementation will follow soon
- ApplicationCore will be the main interface to be used when writing new applications, but some concepts are still missing
- Available on Github / svn:
  `http://github.com/ChimeraTK/ControlSystemAdapter`
  `http://oss.aquenos.com/svnroot/epics-mtca4u`
  `http://github.com/ChimeraTK/ControlSystemAdapter-DoocsAdapter`
  `http://github.com/ChimeraTK/ControlSystemAdapter-OPC-UA-Adapter`
  `http://github.com/ChimeraTK/ApplicationCore`

HELMHOLTZ ASSOCIATION

This work is supported by the Helmholtz Validation Fund HVF-0016 "MTCA.4 for Industry"

FAST LOGIC aquenos software & more