Drivers and Software for MicroTCA.4.

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

MicroTCA.4 Technology

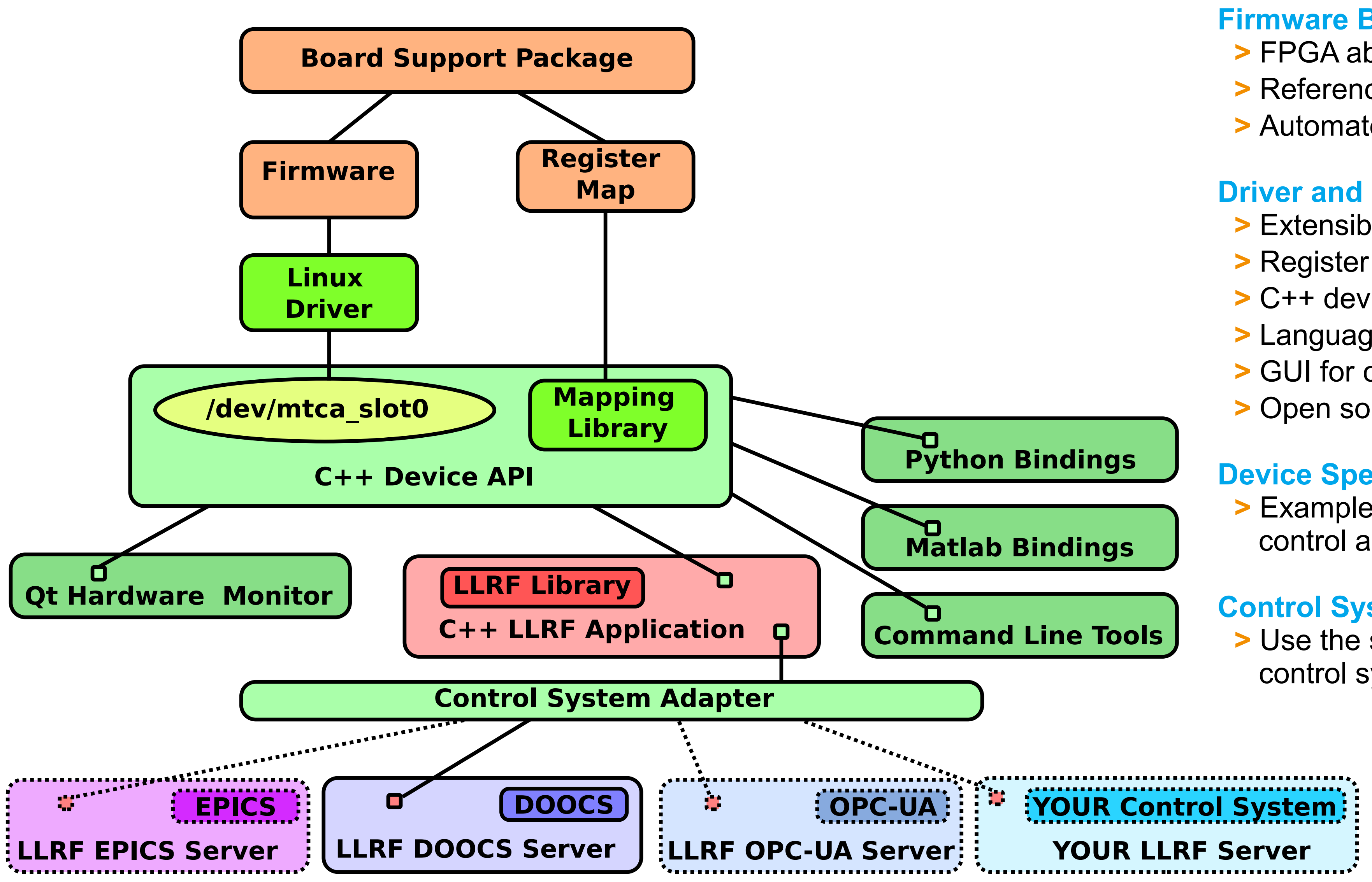
Based on Advanced Telecommunications Computing Architecture (ATCA)
Widespread use in telecommunications since 2005

> High speed serial bus topology
> High modularity due to Advanced Mezzanine Cards (AMCs)
> High availability due to redundancy
> Reduced down-time due to hot-swap capability

MicroTCA.4 Enhancements for Rear I/O and Precision Timing

> Definition of Micro Rear Transition Modules (μRTMs)
> Definition of AMC-μRTM connection
> Radial clock lines for precision timing
> Low latency point to point serial I/O
> Advanced shelf management
> High signal integrity by separation of analog and digital processing

The DESY MicroTCA.4 User Tool Kit (MTCA4U)



Firmware Board Support Package

> FPGA abstraction layer
> Reference firmware with demo application code
> Automated generation of register map

Driver and Basic Tools

> Extensible universal driver
> Register mapping library
> C++ device API
> Language bindings to Matlab and Python
> GUI for convenient register monitoring/setting
> Open source

Device Specific Applications

> Example: Low Level Radio Frequency (LLRF) control application for accelerators

Control System Adapter

> Use the same application code with multiple control systems

Use Case
Low Level Radio Frequency Control at the European XFEL and FLASH

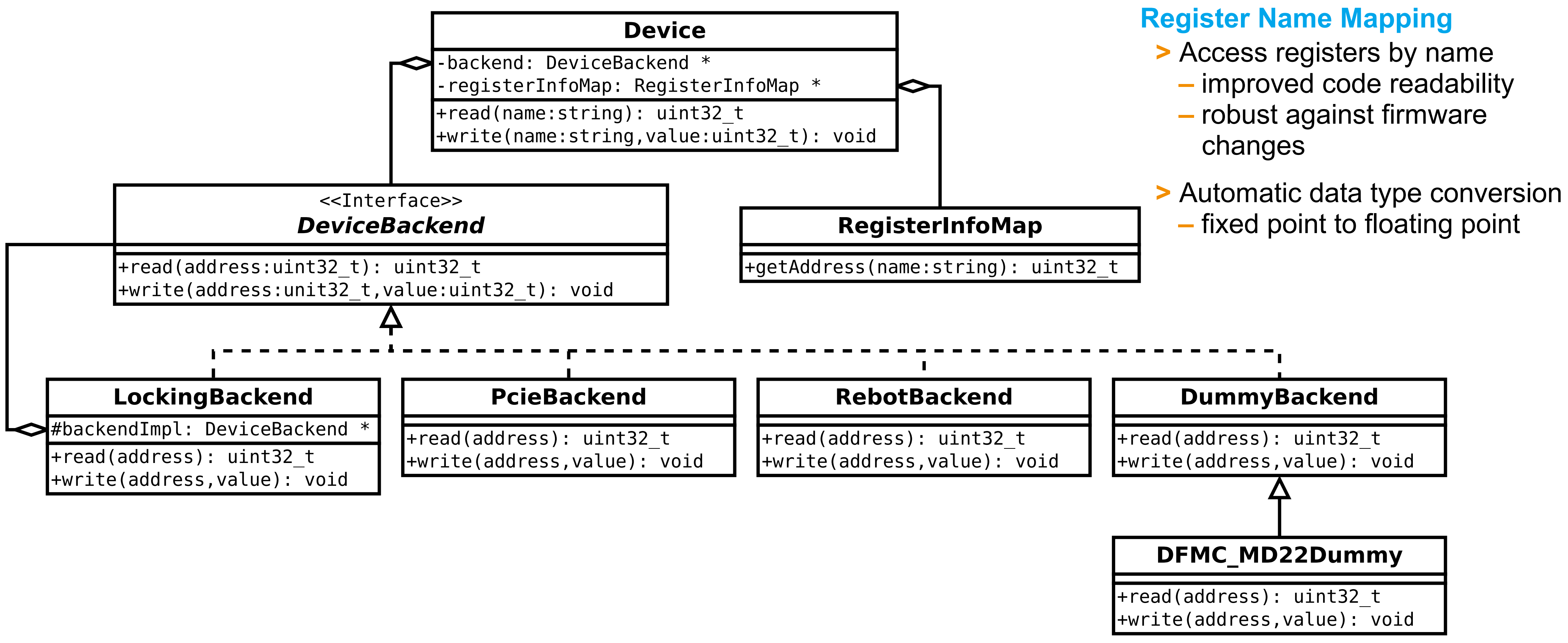
> Superconducting accelerators provide multi-GeV electron beams for Free Electron Lasers (FELs)

> Digital low level radio frequency (LLRF) control based on MicroTCA.4

> Pulsed operation (10 Hz)

MicroTCA.4 LLRF installation in the FLASH accelerator tunnel

The C++ Device API


Device
-backend: DeviceBackend *
-registerInfoMap: RegisterInfoMap *
+read(name:string): uint32_t
+write(name:string,value:uint32_t): void
<<Interface>>
DeviceBackend
+read(address:uint32_t): uint32_t
+write(address:unit32_t,value:uint32_t): void
RegisterInfoMap
+getAddress(name:string): uint32_t
LockingBackend
#backendImpl: DeviceBackend *
+read(address): uint32_t
+write(address,value): void
PcieBackend
+read(address): uint32_t
+write(address,value): void
RebotBackend
+read(address): uint32_t
+write(address,value): void
DummyBackend
+read(address): uint32_t
+write(address,value): void
DFMC_MD22Dummy
+read(address): uint32_t
+write(address,value): void

Register Name Mapping

> Access registers by name
– improved code readability
– robust against firmware changes

> Automatic data type conversion
– fixed point to floating point

DeviceBackend

> Abstract interface

> PCI Express

> Register-based over TCP (ReboT)

Back-End Factory

> Automatically determine the type

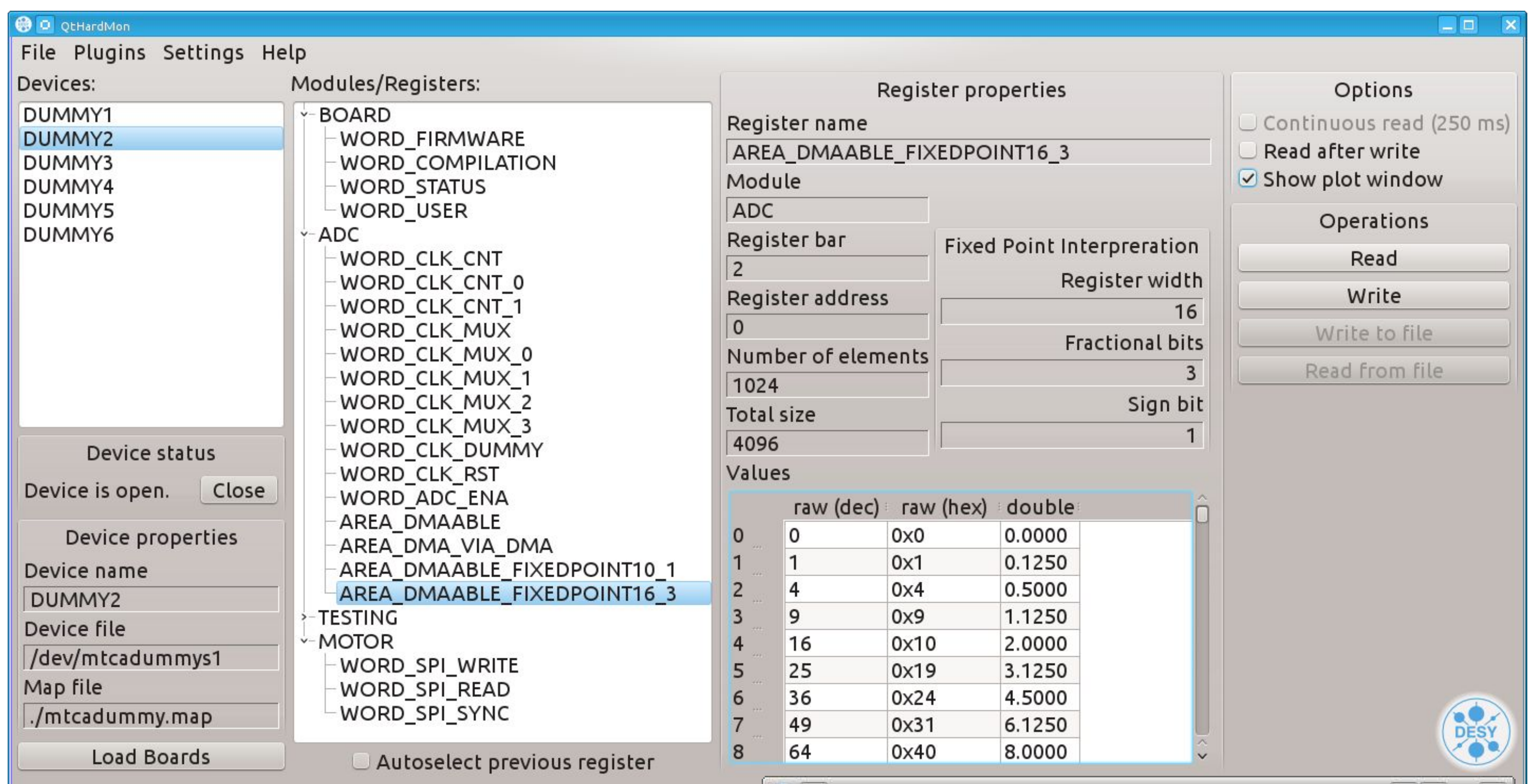
> Plugin mechanism
– add new back-ends at run time

DummyBackend

> Simulate I/O address space in RAM

> Callback functions on read/write
– implement firmware mock-ups

Qt Hardware Monitor



QtHardMon

Devices:
DUMMY1
DUMMY2
DUMMY3
DUMMY4
DUMMY5
DUMMY6

Device status
Device is open.

Device properties
Device name: DUMMY2
Device file: /dev/mtcadummys1
Map file: ./mtcadummy.map

Modules/Registers:
- BOARD
  - WORD_FIRMWARE
  - WORD_COMPILATION
  - WORD_STATUS
  - WORD_USER
- ADC
  - WORD_CLK_CNT
  - WORD_CLK_CNT_0
  - WORD_CLK_CNT_1
  - WORD_CLK_MUX
  - WORD_CLK_MUX_0
  - WORD_CLK_MUX_1
  - WORD_CLK_MUX_2
  - WORD_CLK_MUX_3
  - WORD_CLK_DUMMY
  - WORD_CLK_RST
  - WORD_ADC_ENA
  - AREA_DMAABLE
  - AREA_DMA_VIA_DMA
  - AREA_DMAABLE_FIXEDPOINT10_1
  - AREA_DMAABLE_FIXEDPOINT16_3
- TESTING
- MOTOR
  - WORD_SPI_WRITE
  - WORD_SPI_READ
  - WORD_SPI_SYNC

□ Autoselect previous register

Register properties
Register name: AREA_DMAABLE_FIXEDPOINT16_3
Module: ADC
Register bar: 2
Register address: 0
Number of elements: 1024
Total size: 4096

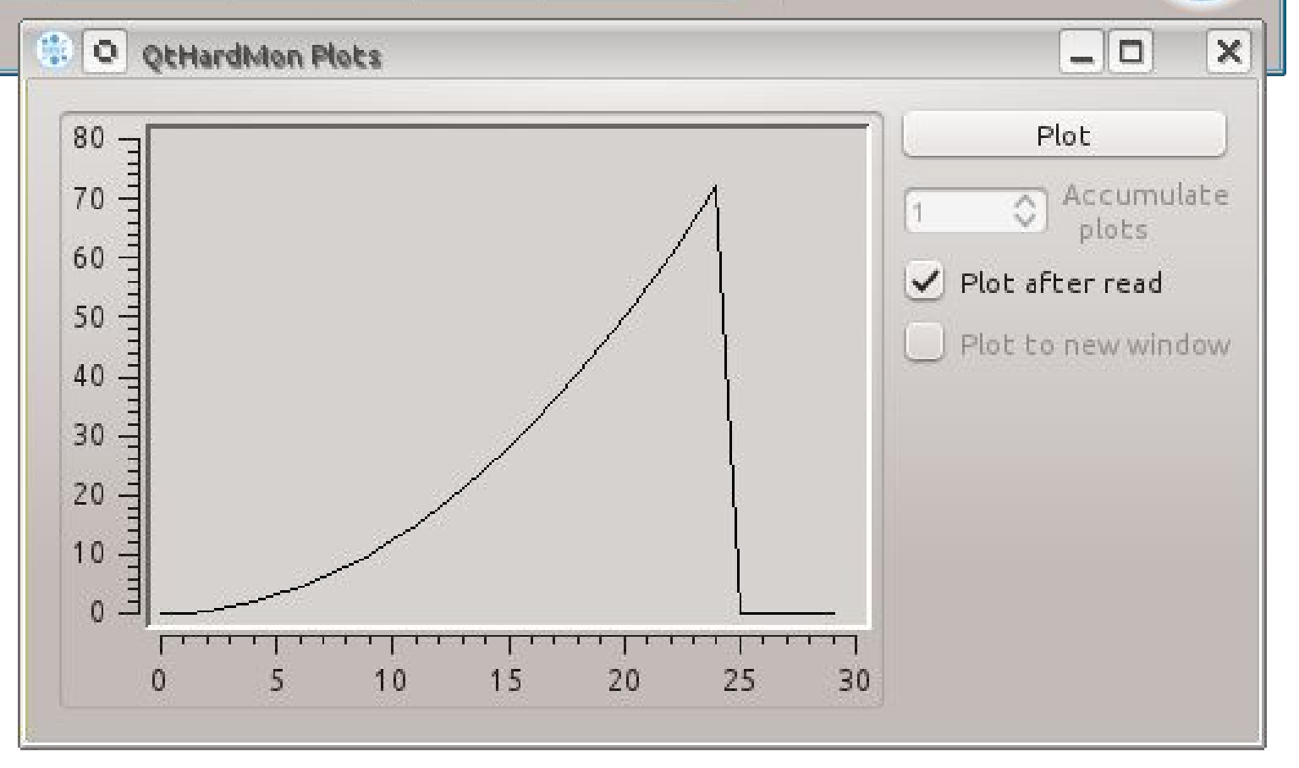
Fixed Point Interpreration
Register width: 16
Fractional bits: 3
Sign bit: 1

Values

	raw (dec)	raw (hex)	double
0	0	0x0	0.0000
1	1	0x1	0.1250
2	4	0x4	0.5000
3	9	0x9	1.1250
4	16	0x10	2.0000
5	25	0x19	3.1250
6	36	0x24	4.5000
7	49	0x31	6.1250
8	64	0x40	8.0000

Options
□ Continuous read (250 ms)
□ Read after write
☑ Show plot window

Operations



QtHardMon Plots

1 Accumulate plots
☑ Plot after read
□ Plot to new window

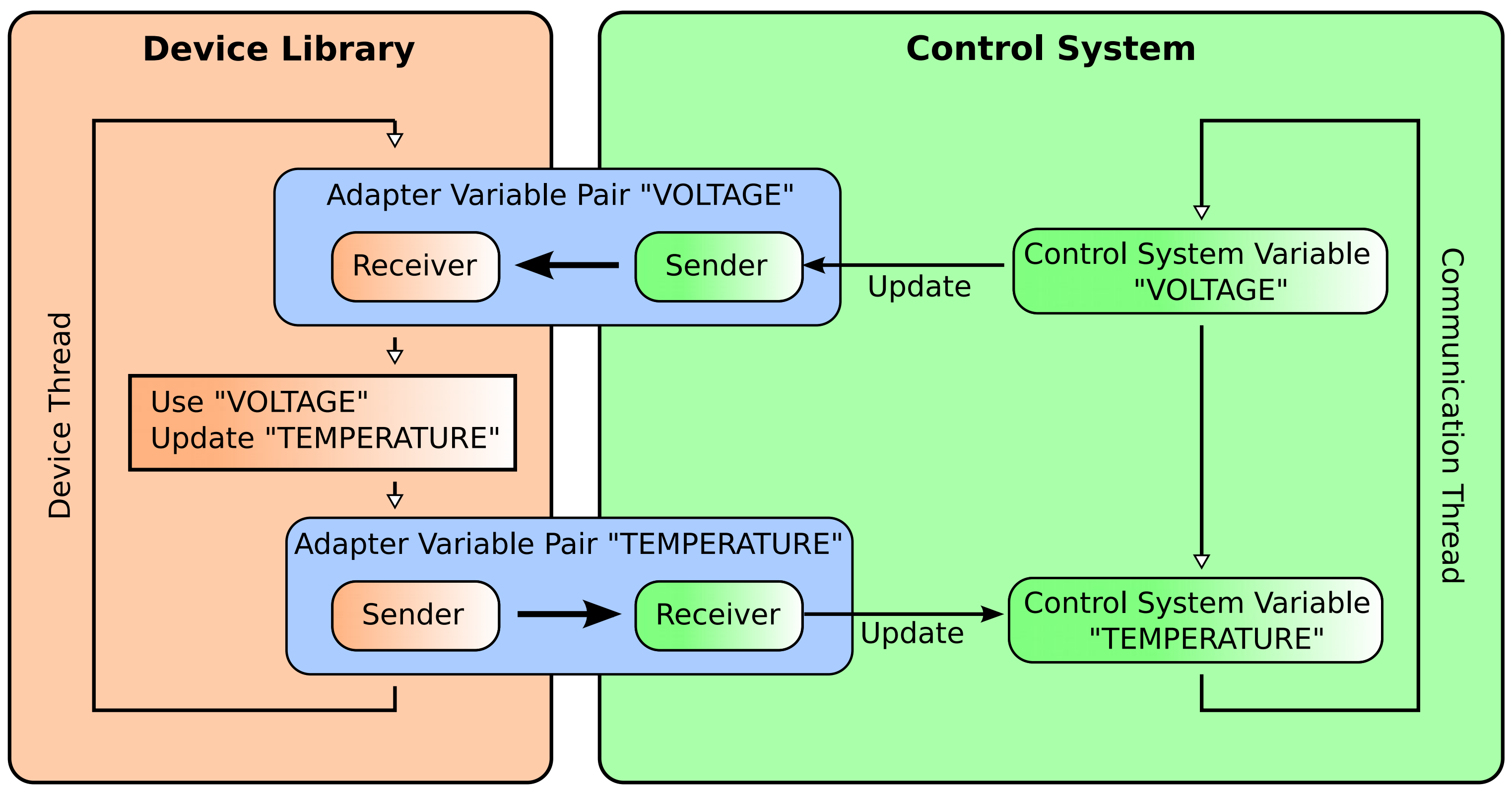
> Lists all hardware registers

> Register names and properties

> Read and modify register content

> Basic plotting functionality

Control System Adapter


Device Library
Control System
Adapter Variable Pair "VOLTAGE"
Receiver
Sender
Update
Control System Variable "VOLTAGE"
Use "VOLTAGE"
Update "TEMPERATURE"
Adapter Variable Pair "TEMPERATURE"
Sender
Receiver
Update
Control System Variable "TEMPERATURE"
Device Thread
Communication Thread

Task

> Keep the application code (device library) independent from the control system

> Minimise device-dependent code on the control system side

Requirements

> Thread safety

> Real-time capability

> Do not copy large data objects

Subversion Repository: https://svnsrv.desy.de/public/mtca4u

HELMHOLTZ | ASSOCIATION

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

μTCA®

ICALEPCS 2015
Melbourne, Australia

Presenter: Martin Killenberg
e-mail: martin.killenberg@desy.de

FAST LOGIC

aquenos
software & more

COSYLAB