# BROADEN HORIZONS.

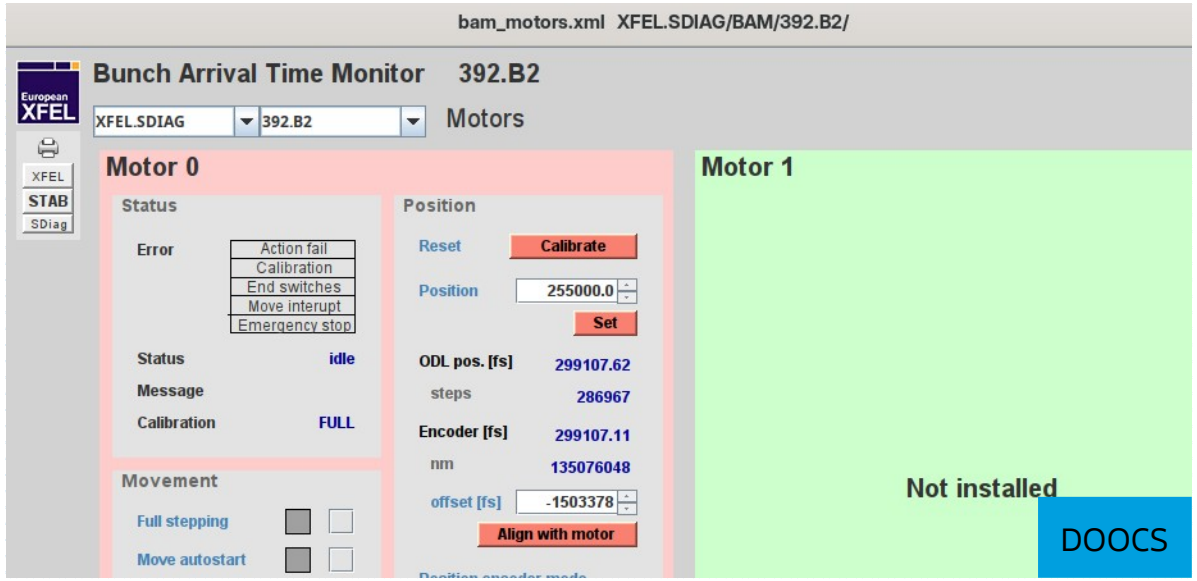

## Opening an existing software portfolio to a TANGO environment

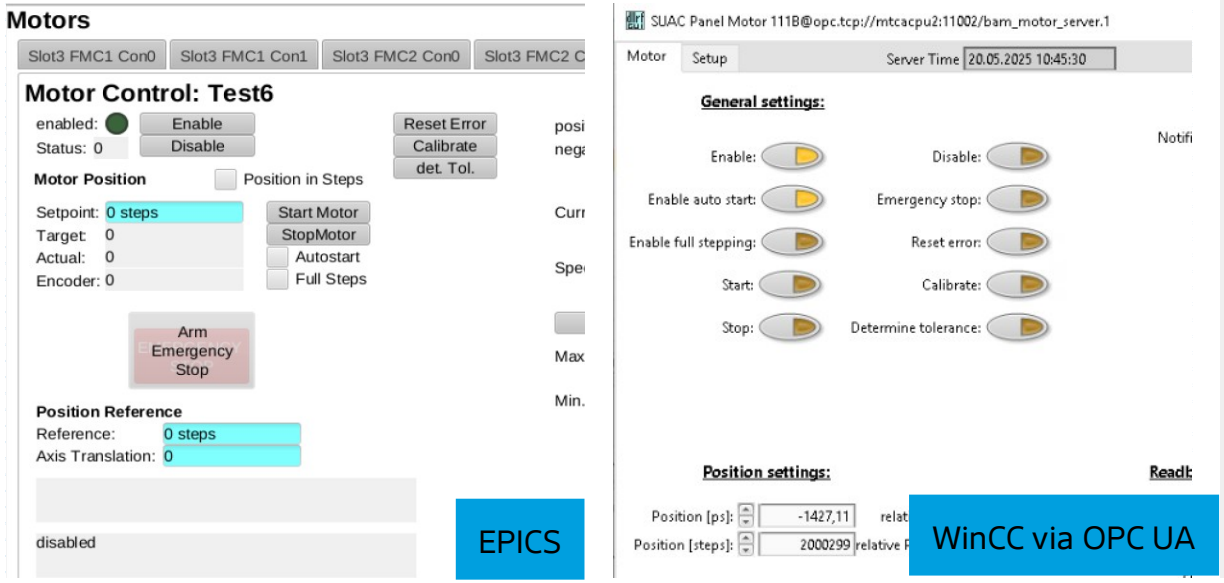
Jens Georg
Software Engineer, MSK, DESY

Hamburg/Giulianova, 2025-05-22


HELMHOLTZ

# One Software:
# Four control systems


bam_motors.xml XFEL.SDIAG/BAM/392.B2/
Bunch Arrival Time Monitor 392.B2
XFEL.SDIAG
392.B2
Motors
Motor 0
Status
Error
Action fail
Calibration
End switches
Move interupt
Emergency stop
Status idle
Message
Calibration FULL
Movement
Full stepping
Move autostart
Position
Reset
Calibrate
Position 255000.0
Set
ODL pos. [fs] 299107.62
steps 286967
Encoder [fs] 299107.11
nm 135076048
offset [fs] -1503378
Align with motor
Motor 1
Not installed
DOOCS


Motors
Slot3 FMC1 Con0
Slot3 FMC1 Con1
Slot3 FMC2 Con0
Motor Control: Test6
enabled:
Enable
Disable
Status: 0
Reset Error
Calibrate
det. Tol.
Motor Position
Position in Steps
Setpoint: 0 steps
Target: 0
Actual: 0
Encoder: 0
Start Motor
StopMotor
Autostart
Full Steps
Arm Emergency Stop
Position Reference
Reference: 0 steps
Axis Translation: 0
disabled
EPICS
SUAC Panel Motor 111B@opc.tcp://mtcacpu2:11002/bam_motor_server.1
Motor
Setup
Server Time 20.05.2025 10:45:30
General settings:
Enable:
Disable:
Enable auto start:
Emergency stop:
Enable full stepping:
Reset error:
Start:
Calibrate:
Stop:
Determine tolerance:
Position settings:
Position [ps]: -1427,11
Position [steps]: 2000299
WinCC via OPC UA


AtkPanel 5.9 : test/md22/0
View Preferences Help
test/md22/0
test/md22/0
Application is running.
Trigger_period 100 n/a
Motors_nMotors 1 n/a
Motors_motorDriverType MD22
Motors_motorType LinearMotorWithReferenceSwitch
Motors_motorDriverDeviceName MotorDriver1
Motors_motorDriverCardName BSP
Motors_motorDriverModuleName FMC2
Motors_motorDriverId 0 n/a
Motors_motorDriverConfigFile Limes122-MotorDriverCardConfig.xml
Motors_userUnitToStepsRatio 0.00 n/a
Motors_encoderUnitToStepsRatio 0.00 n/a
Motors_userPositionUnit mm
Motors_dummy 0 n/a
Motor1.start 0 n/a
Motor1.emergencyStop 0 n/a

# Tangent:
# How did we end up here

# Who is DESY MSK?

## Overview of our tasks

**What we do.**

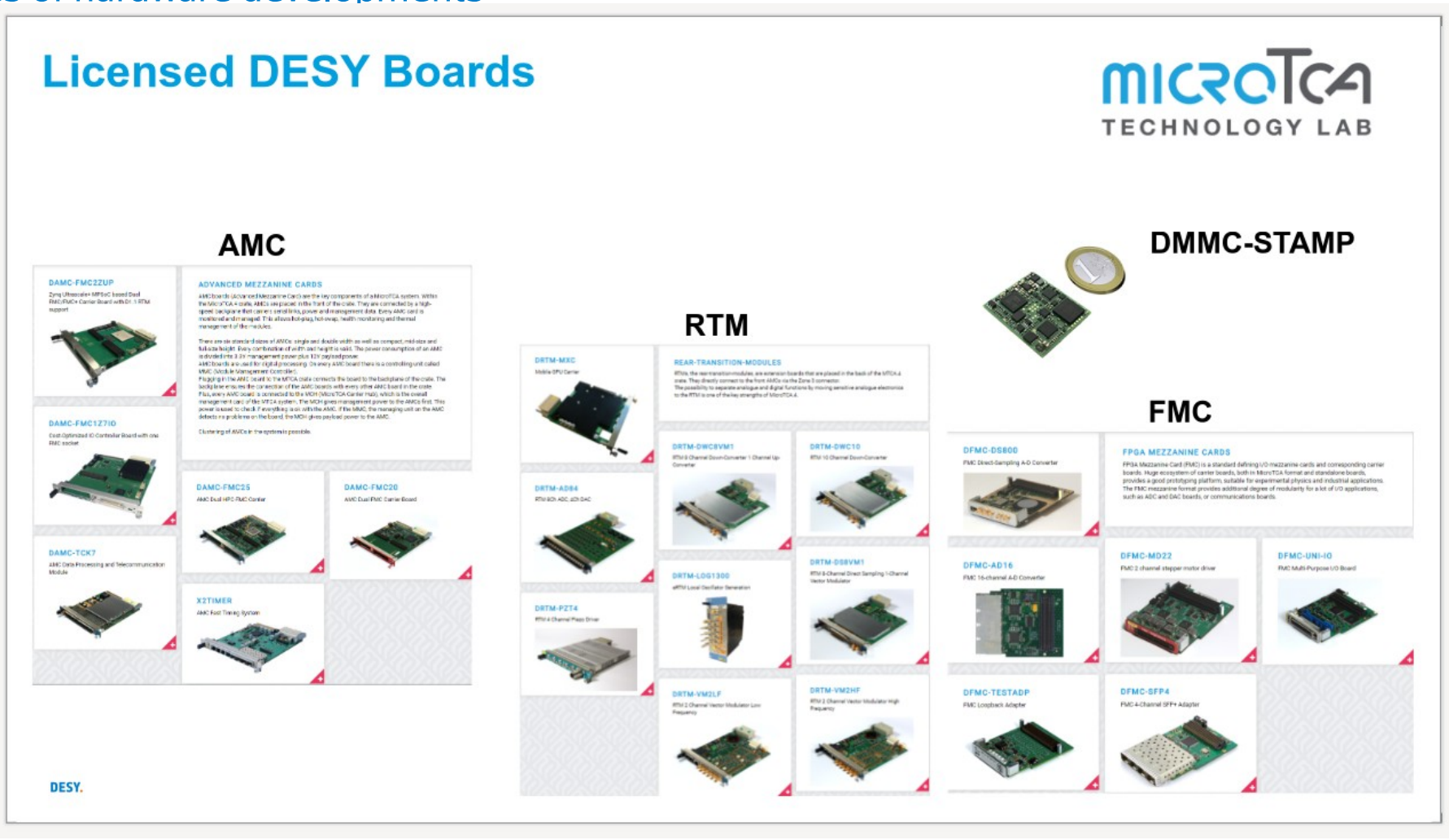
- Beam Controls group in DESY's machine division
- M division runs the accelerators such as EuXFEL, FLASH, PETRA III
- We do LLRF, feedback, synchronisation, special diagnostics and intelligent process control
- Develop our own own analogue and digital hardware, mainly for µTCA systems
- Write firmware and software for using those devices

**Challenges.**

- Developments done in collaborations within and outside of Helmholtz
- Confronted with a zoo of systems: DOOCS, EPICS, WinCC, ...
- How to integrate our hard- and software with such heterogeneous environments?
- Possible solution: Write shared libraries, re-write control system integration for each server
  - Tedious, repetitive work

# Hardware

## Some examples of hardware developments



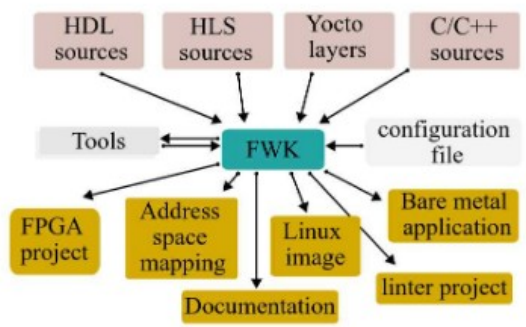
# FPGA Firmware

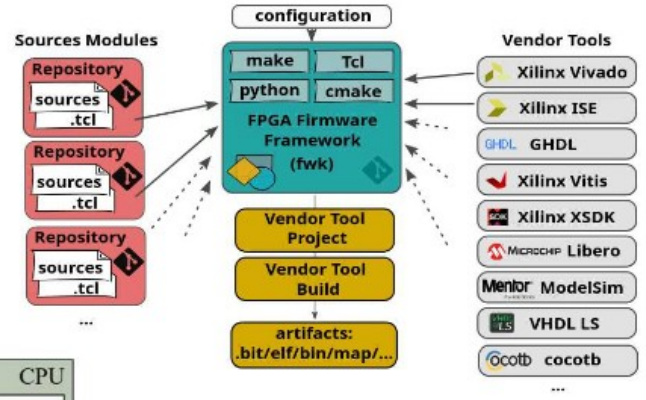
## DESY.MSK.FWK

### An Open Source Firmware Development Framework

FWK
FPGA firmware framework

repo: gitlab.desy.de/fpgafw/fwk
doc : fpgafw.pages.desy.de/docs-pub/fwk




THE DESY OPEN SOURCE FPGA FRAMEWORK

L. Butkowski*, A. Bellandi, B. Dursun, C. Gümüs, K. Schulz,
M. Büchler, N. Omidsajedi, Deutsches Elektronen-Synchrotron DESY, Hamburg, Germany

MO4AO03

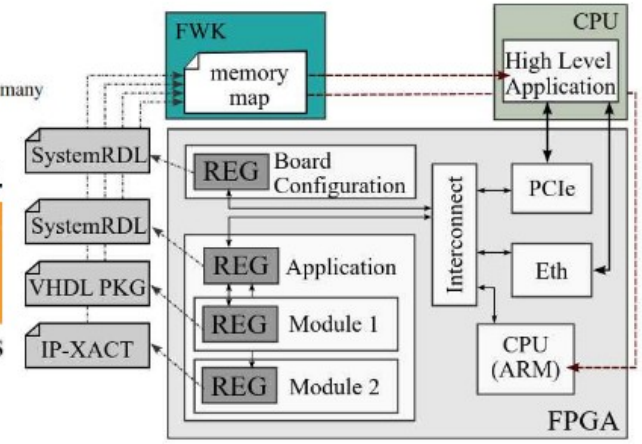
ICALEPCS CAPE TOWN SOUTH AFRICA 2023

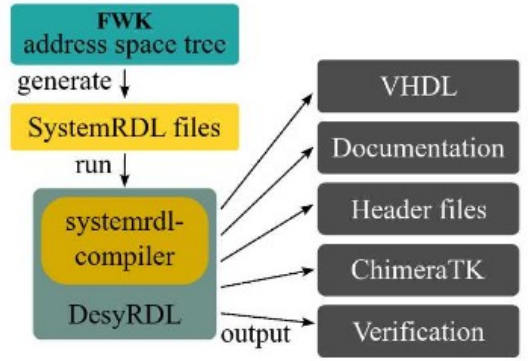
WEPGF074

ICALEPCS 2015
International Conference on Accelerator & Large Experimental Physics Control Systems

FPGA FIRMWARE FRAMEWORK FOR MTCA.4 AMC MODULES

Lukasz Butkowski, Tomasz Kozak, Bin Yang, DESY, Hamburg, Germany
Paweł Prędki, DMCS, Lodz University of Technology, Lodz, Poland
Radoslaw Rybaniec, ISE, Warsaw University of Technology, Warsaw, Poland




DESY. | Burak Dursun | Model Based Verification | LLRF Workshop | Gyeongju, Republic of Korea, 2023-10-26 14

# ChimeraTK

## Three columns of abstraction and integration

**DeviceAccess**

- Client access to hardware devices and other control system applications
- Supports PCIe, UIO, MODBUS, “chat-based devices” (SCPI), DOOCS, EPICS, OPC UA, TANGO,...
- Logical devices
- Triggered device read-out (timing, periodic, interrupts...)
- Integrates with DESY-FWK

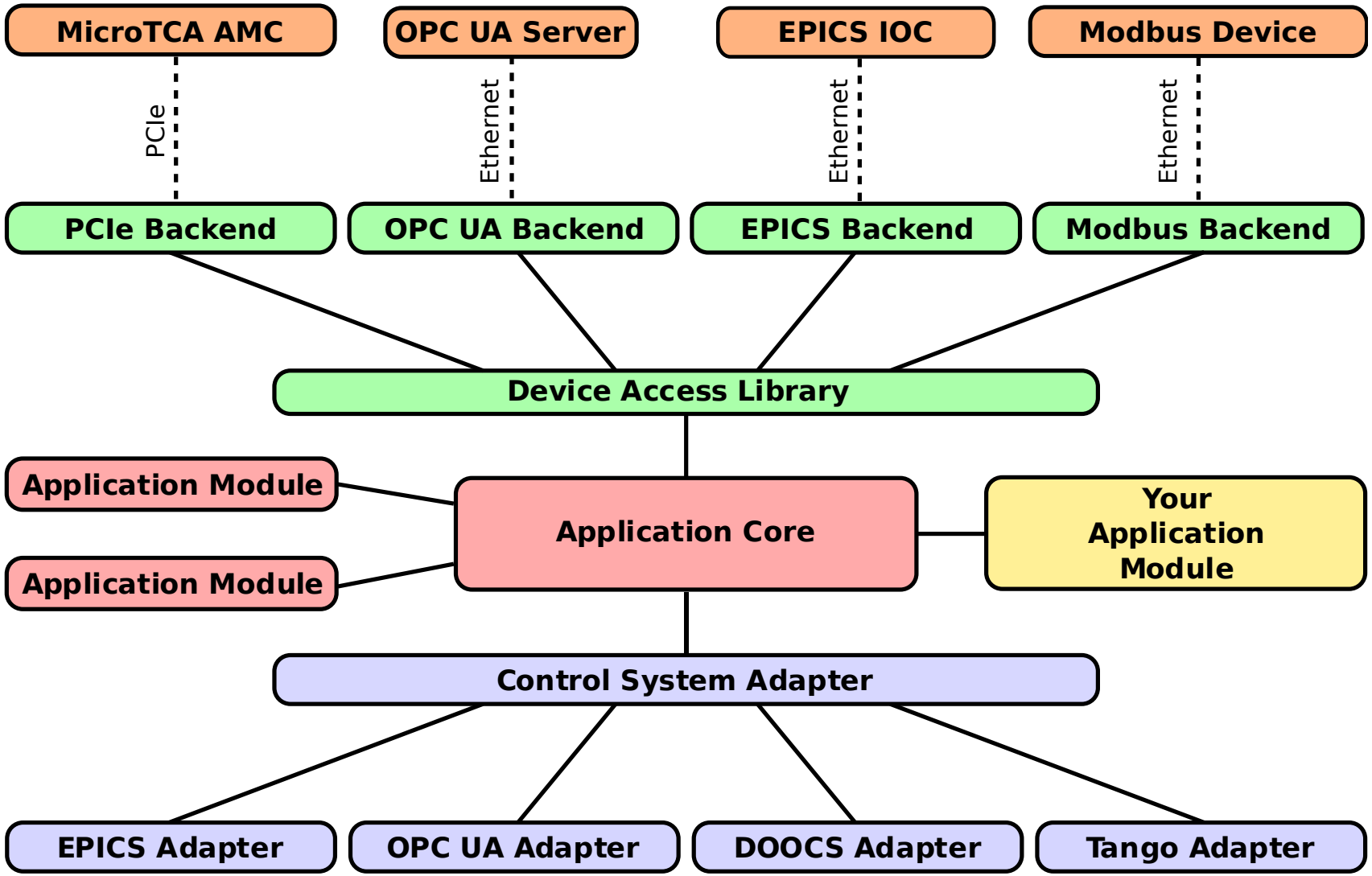
**ApplicationCore**

- Application logic written independently of concrete hardware or control system integration
- Use of small blocks of functionality (Modules) connected together
- Automatic recovery from device failures, data validity handling
- Python scripting support

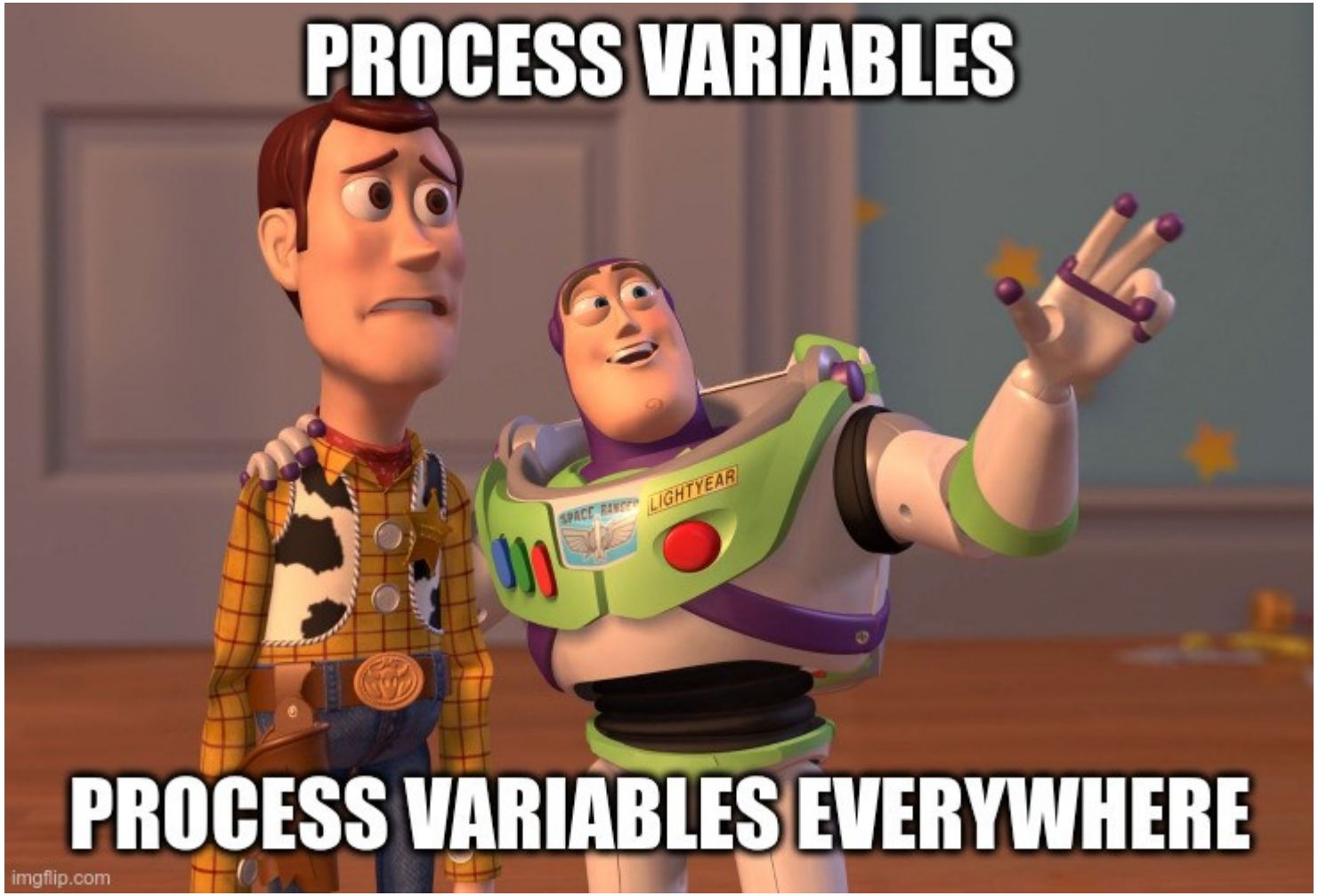
**ControlSystemAdapter**

- Translation layer from ApplicationCore to target control system framework
- Supports DOOCS, EPICS, TANGO and OPC UA
- Highly configurable to adapt to the required environment

# ChimeraTK Overview



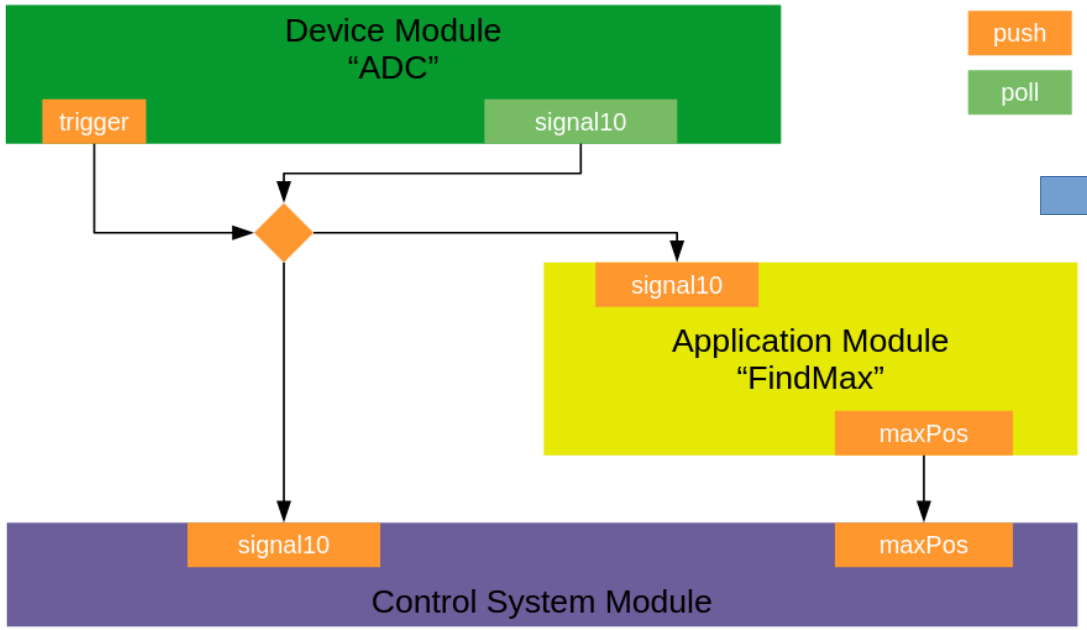
# Application Modules



# Application Modules

## Extend servers with business logic

### ApplicationModule concept




**Register = Process Variable = CS Property**

- New business logic as ApplicationModule
- Self-contained
- Runs in its own thread
- Modern inter-thread communication with lock-free queues
- Connection code is automatically generated

# ControlSystemAdapter

## Talking to the rest of the world

**A rough overview of control system framework integration.**

- No change in application logic required
- The adapter has to be selected at **compile** time
  - Preventing incompatibility issues
  - Limitations posed by some control system frameworks
- Adapters written such that they can work without additional mapping
  - With exceptions such (EPICS, DOOCS)
  - Usually one wants mapping to reduce and rename process variables
- Additional configuration necessary for the target control system
  - Registering devices in JIVE
  - DOOCS server configuration file
  - ….

# Asking to Dance: Diving into TANGO

- TANGO integration has been on our feature request list for years.
- Became urgent 2023/2024 when hardware collaboration with DESY FS was on the horizon

# TANGO ControlSystemAdapter

## Overview of our tasks

**How we got there.**

- Huge contribution from SOLEIL, by Jade Pham
- Has been extended to allow more flexible configuration
- Supports scalar and spectrum attributes
- Can map subtrees from ApplicationCore into devices
- Supports multiple device classes per device server

**What needs to be done.**

- Support of server-side events: CHANGE and DATA_READY
- Support for commands
- Image support
- Allow a more TANGO-like experience, like being able to re-initialize the device server on the fly

# TANGO example configuration

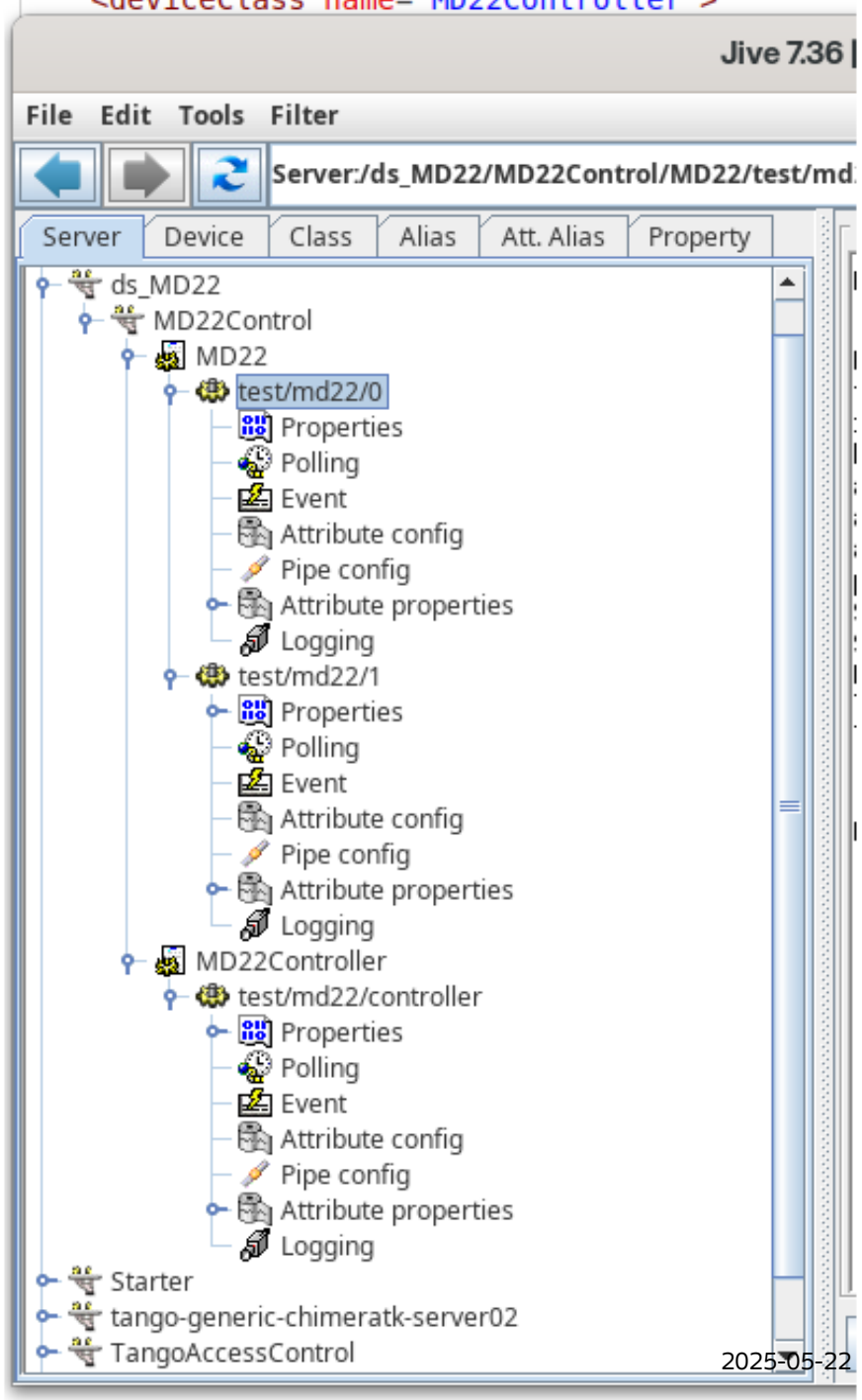
D22-AttributeMapper.xml ●

D22-AttributeMapper.xml > ...

```xml
<?xml version="1.0" encoding="UTF-8"?>
<deviceServer>
    <deviceClass name="MD22">
        <description>DFMC-MD22 stepper motor</description>
        <deviceInstance name="test/md22/0">
            <import>/Motor1</import>
        </deviceInstance>
        <deviceInstance name="test/md22/1">
            <import>/Motor2</import>
        </deviceInstance>
    </deviceClass>
    <deviceClass name="MD22Controller">
        <title>MD22 Controller</title>
        <description>DFMC-MD22 stepper controller</description>
        <deviceInstance name="test/md22/controller">
            <attribute name="numberOfMotors" source="/Motors/nMotors">
                <description>The number of motors in this device server</description>
                <egu>n/a</egu>
            </attribute>
        </deviceInstance>
    </deviceClass>
</deviceServer>
```

Features from the TANGO configuration

- Multiple device classes in one server
- Attaching devices to a device class, mapping them to parts of the hierarchy from the application
- Manual mapping of properties, overriding descriptions and egu
- Works without configuration, will map all variables into the first device it is requested to add. DeviceClass name will be derived from executable name.




# TANGO example configuration

Result in Jive from the previous configuration.

- Devices not configured in the mapper will be skipped by the server!

# TANGO DeviceAccess Backend

## Current state and outlook

**What is implemented.**

- Read/write of attributes
- Data quality transport into the framework

**What needs to be done.**

- Support of server-side events: CHANGE and DATA_READY
- Support of READ_WITH_WRITE
- Image support
- Strategy for command handling
- More tests against real life TANGO device servers

# Links

## Documentation and other presentations around FWK and ChimeraTK


**Further references.**

- Open Source: FWK is Apache 2.0 for the framework,  CERN-OHL-W-2.0 for VHDL modules.
- ChimeraTK is LGPL-3.0-or-later
- Packages for Bookworm and Ubuntu on DOOCS package repository: https://doocs-web.desy.de/pub/doocs
- Yocto layer: https://github.com/ChimeraTK/meta-chimeratk

- DESY FWK documentation:  https://fpgafw.pages.desy.de/docs-pub/fwk
- DESY FWK repository: https://gitlab.desy.de/fpgafw/fwk
- ChimeraTK tutorial: https://indico.desy.de/event/46036/contributions/178921/attachments/94226/128412/chimeratk-tutorial.pdf
- ChimeraTK at github: https://github.com/ChimeraTK/
- ChimeraTK documentation hub: https://chimeratk.github.io/

# Thank You

**Contact**

Jens Georg
E-Mail: jens.georg@desy.de

**Deutsches Elektronen-Synchrotron DESY**

MSK
Notkestraße 85
22607 Hamburg
www.desy.de

# Bonus: Low-code hardware integration

## GenericDeviceServer

### Often, device integration is just read-out and configuration

Initialise the device, accept  some parameters from user, provide data to control system.

### Queue the the GenericDeviceServer

It is available for every control system framework. With this, the device integration is usually just a matter of writing a couple of configuration files. It is used “in production” at DESY for controlling the Main Oscillator in FLASH or SOLEIL’s fast orbit feedback system (on embedded Linux). Application logic can be added through Python.