Introduction to DESY's firmware framework FWK

An open-source FPGA framework by DESY for large scientific projects

Michael Randall
On behalf of the DESY MSK Firmware Team
Chongqing, 17th of September

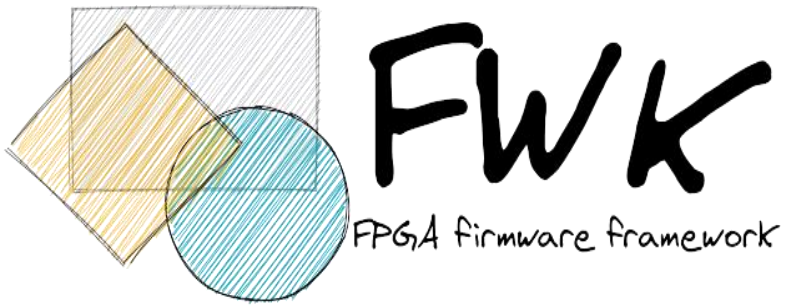
HELMHOLTZ

DESY.

MSK Firmware Development

Background

- FPGA firmware team from the DESY Accelerator Beam Controls group (MSK)
- FLASH and European XFEL facilities, upcoming PETRA IV, and many more
- Various firmware projects:
  - Low-Level RF
  - Beam Arrival Monitor
  - Bunch Compression Monitor
  - Laser-based Synchronization
  - …
- Majority based on MTCA.4 hardware
- Streamlined development through open-source FPGA Firmware Framework (FWK)

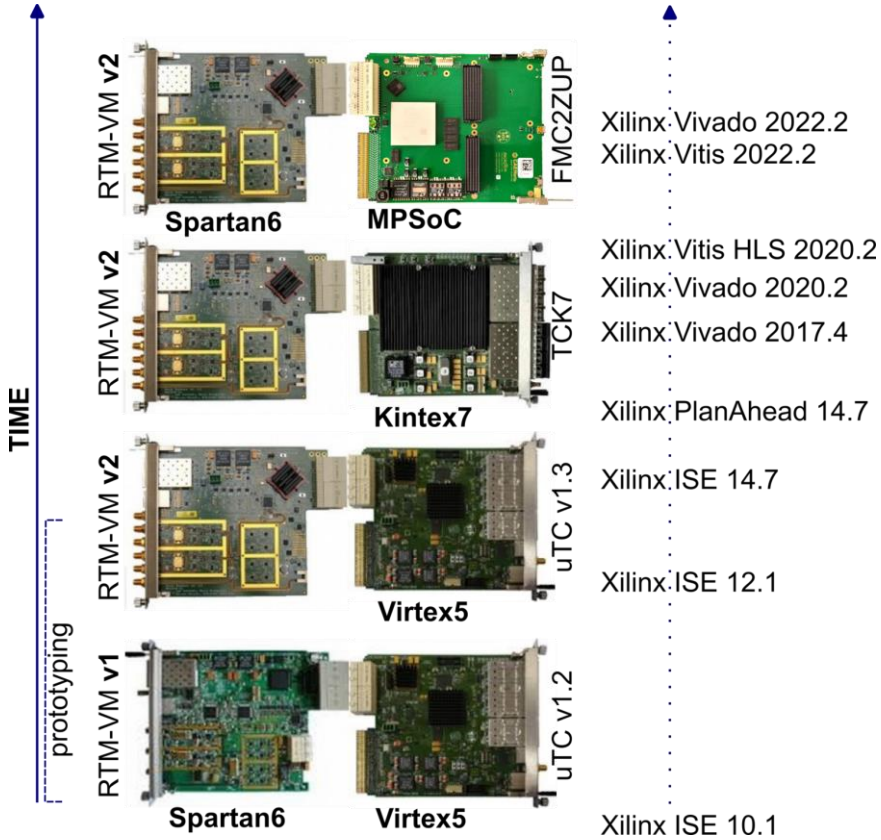


MSK Firmware Development

Hardware & Application Complexity

Same application – different hardware – different tool

e.g.: EuXFEL LLRF controller example on MTCA.4



Same FPGA board – same application - different use case

e.g.: Single Cavity Controller

DWC8VM1

SIS8300L2

10 different configurations

Same FPGA board – different application

- LLRF field detection
- Laser lock controller
- Single Cavity GDR controller
- Single Cavity SEL controller
- Bunch Compressor Monitors

DWC8VM1

DWC10

SIS8900

SIS8300L2

MSK Firmware Development

Development & Maintenance Challenges

- Large number of projects, small core team
- Multiple developers and collaboration involved
- Multiple solution domains:
  - FPGA logic (VHDL, HLS, ...)
  - Bare-metal software (C/C++)
  - Embedded OS (FreeRTOS, Linux)
  - Applications (C++, Python, ...)
- Long-term support and maintenance
  - Evolving projects and technologies
  - Legacy projects

Requested

Goals of FWK:
- Streamline workflows
- Reduce development time & complexity
- Increase collaboration & re-usability

FPGA Firmware Framework (FWK)

Key Features

History:
- Started in 2013 (monolithic code base, still SVN)
- **Open source** 2022 (Apache 2.0 license)

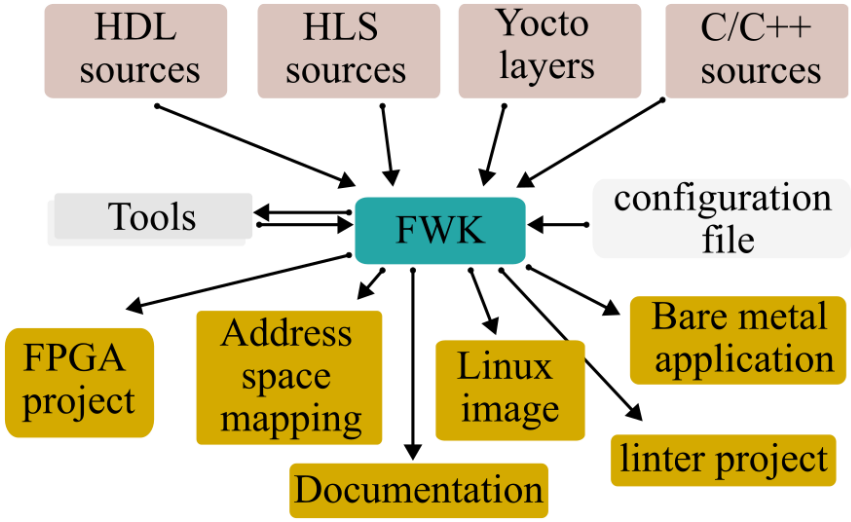
Abstraction & Modularity:
- Modular project structure (**APP & BSP**)
- Tool interaction abstracted
- Standardized development workflow

Supported Firmware Domains:
- FPGA logic design (HDL, HLS, …)
- Verification (GHDL, ModelSim, …)
- Bare-metal software development (C/C++)
- Embedded OS generation (Linux, RTOS)

Address-space Management:
- SystemRDL descriptions
- DesyRDL: code & doc gen
- Integration with ChimeraTK

Reproducible Builds:
- Consistent results
- Allows automation and CI-friendly

Docs-as-Code:
- Documentation located with code
- Generated with FWK



Framework concept

FPGA Firmware Framework (FWK)

FWK as an Orchestrator

Major FWK Components:

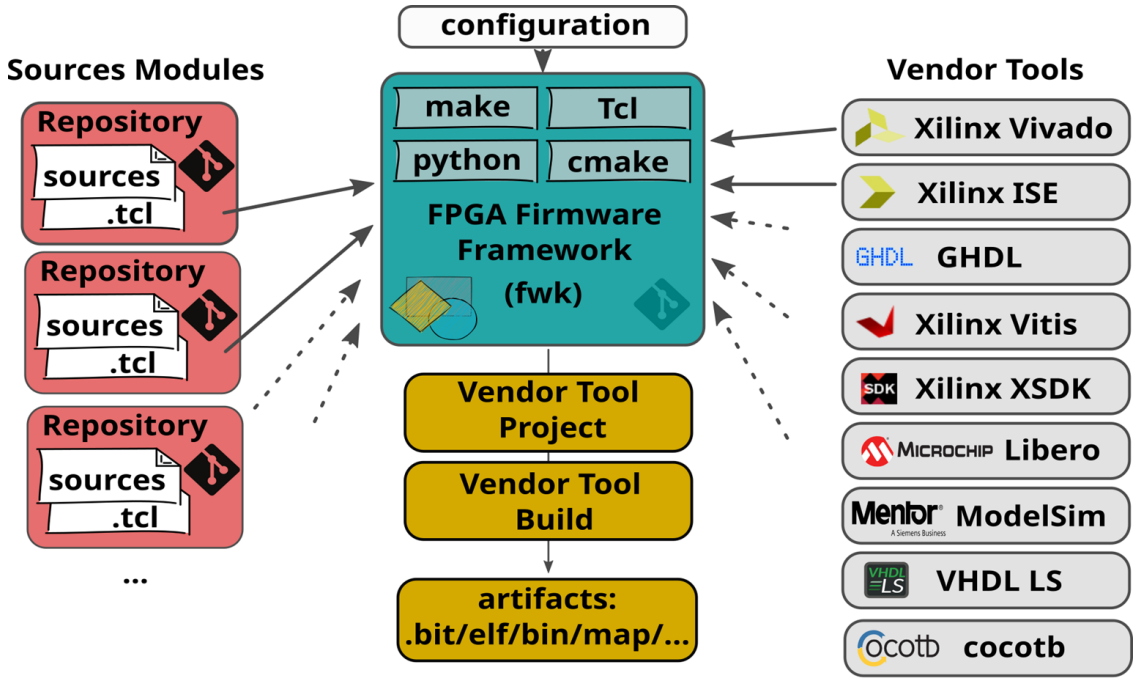
- Framework module (fwk)*
- Source modules*
- Main project*
- Vendor tools

GNU Make as the entry point:

```
make cfg=<config> project [gui] [sim] [build] [doc]
```

Tool Command Language (Tcl):

- Orchestration and vendor tool interaction
- Source module configuration files



Project overview

* Separate Git repositories

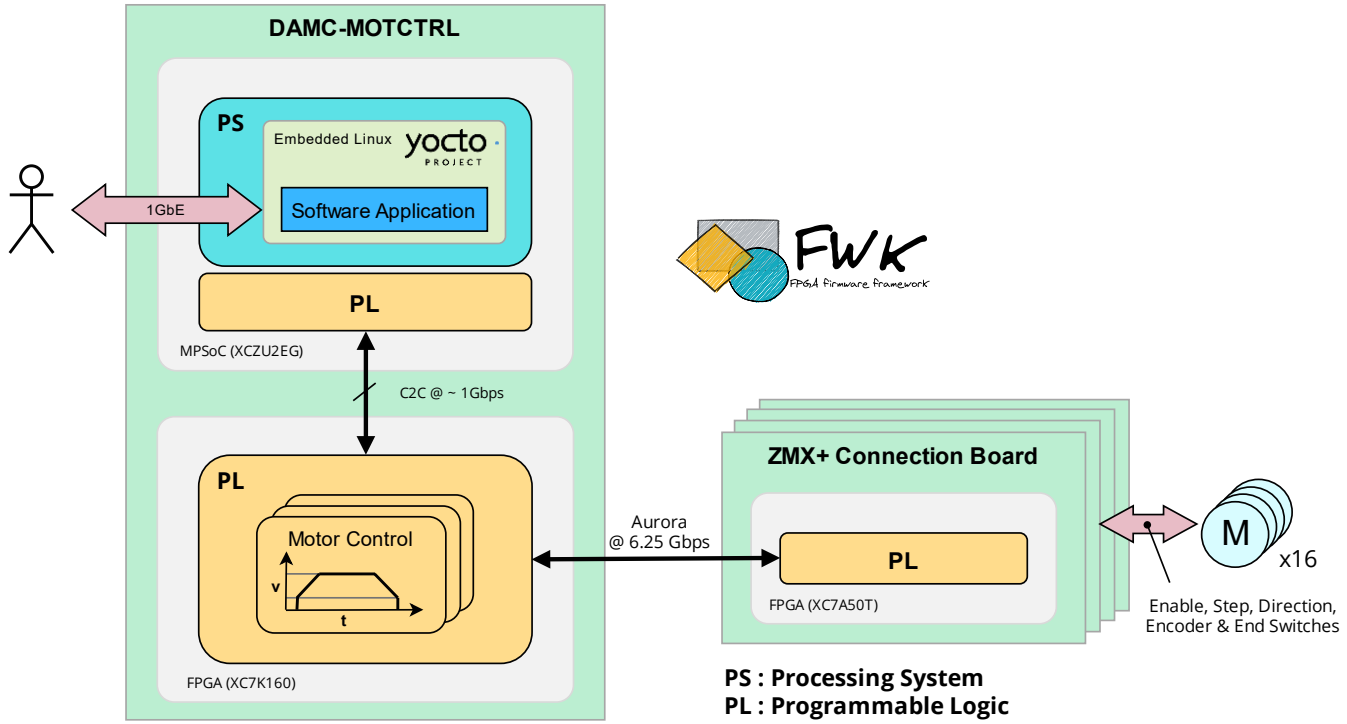
FPGA Firmware Framework (FWK)

Workflow on a Real Project

```
# clone repository
    git clone git@ ... && cd <repo>

# clone all submodules repositories
    git submodules update --init --recursive

# create virtual environment with tools
    make env

# source the Vivado environment
    source /opt/Xilinx/Vivado/2020.2/settings64.sh
```

The PETRA IV Motion Controller* firmware stack:

- 3 x FPGA designs
- Customized Linux image + boot loader
- C++ application software



*See talk MicroTCA-based Motion Controller for more

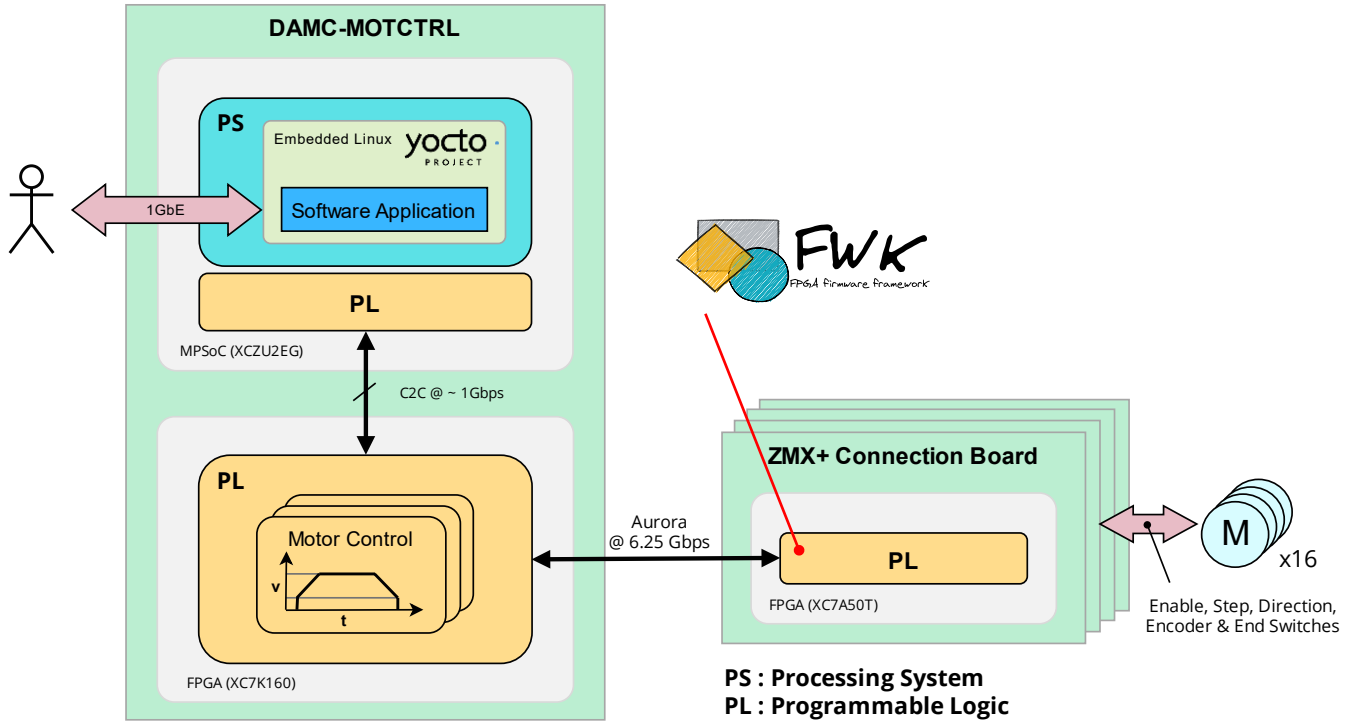
FPGA Firmware Framework (FWK)

Workflow on a Real Project

```
# clone repository
    git clone git@ ... && cd <repo>

# clone all submodules repositories
    git submodules update --init --recursive

# create virtual environment with tools
    make env

# source the Vivado environment
    source /opt/Xilinx/Vivado/2020.2/settings64.sh

# build ZMX+ Connection Board firmware
    make cfg=default_zmx project [gui] build
```

The PETRA IV Motion Controller* firmware stack:

- 3 x FPGA designs
- Customized Linux image + boot loader
- C++ application software



*See talk MicroTCA-based Motion Controller for more

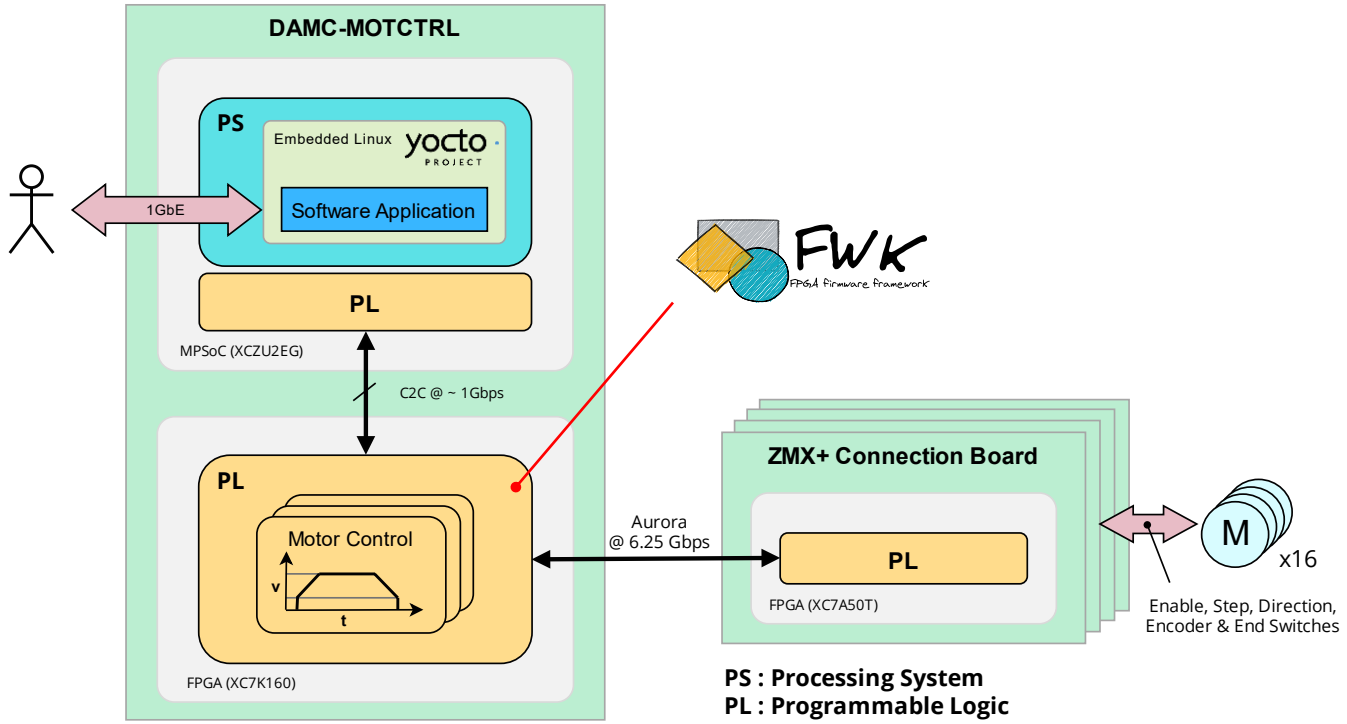
FPGA Firmware Framework (FWK)

Workflow on a Real Project

```
# clone repository
    git clone git@ ... && cd <repo>

# clone all submodules repositories
    git submodules update --init --recursive

# create virtual environment with tools
    make env

# source the Vivado environment
    source /opt/Xilinx/Vivado/2020.2/settings64.sh

# build ZMX+ Connection Board firmware
    make cfg=default_zmx project build

# build DAMC-MOTCTRL (Kintex) firmware
    make cfg=default_k7 project build
```

The PETRA IV Motion Controller* firmware stack:

- 3 x FPGA designs
- Customized Linux image + boot loader
- C++ application software



*See talk MicroTCA-based Motion Controller for more

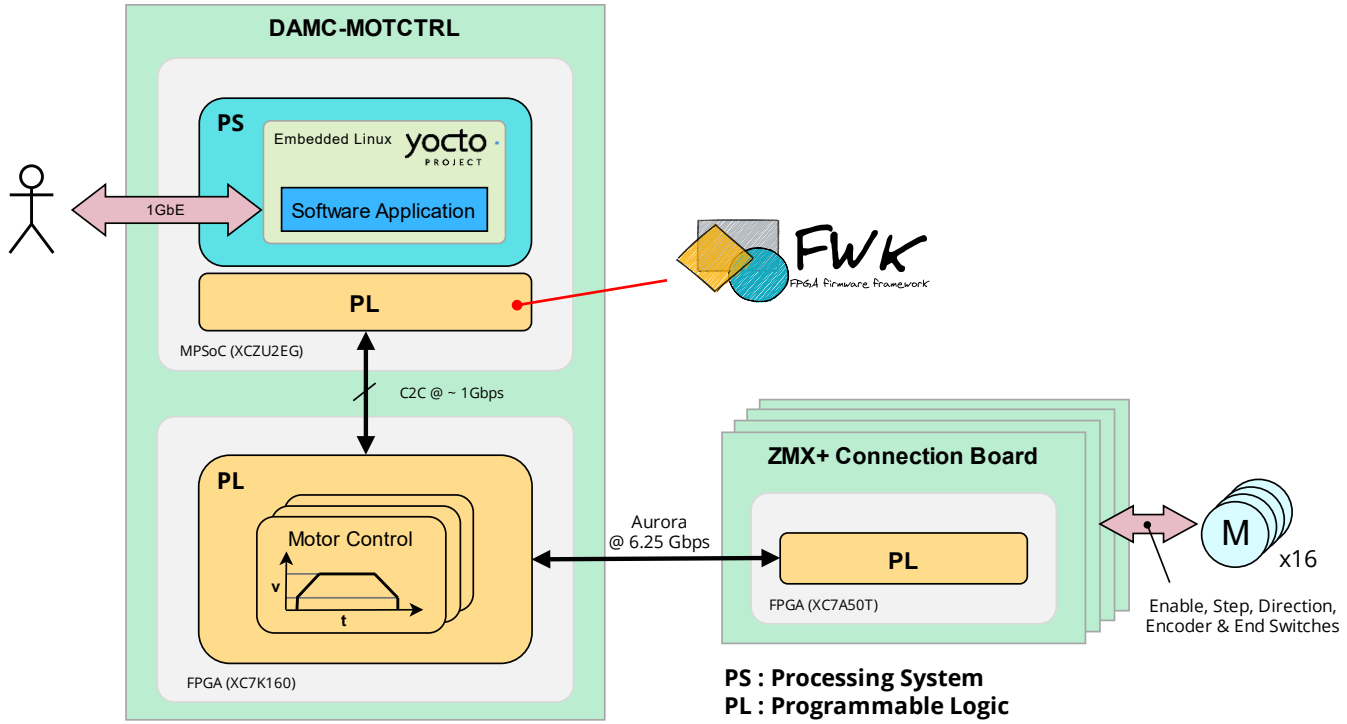
FPGA Firmware Framework (FWK)

Workflow on a Real Project

```
# clone repository
    git clone git@ ... && cd <repo>

# clone all submodules repositories
    git submodules update --init --recursive

# create virtual environment with tools
    make env

# source the Vivado environment
    source /opt/Xilinx/Vivado/2020.2/settings64.sh

# build ZMX+ Connection Board firmware
    make cfg=default_zmx project build

# build DAMC-MOTCTRL (Kintex) firmware
    make cfg=default_k7 project build

# build DAMC-MOTCTRL (Zynq UltraScale+) PL firmware
    make cfg=default_us project build
```

The PETRA IV Motion Controller* firmware stack:

- 3 x FPGA designs
- Customized Linux image + boot loader
- C++ application software



*See talk MicroTCA-based Motion Controller for more

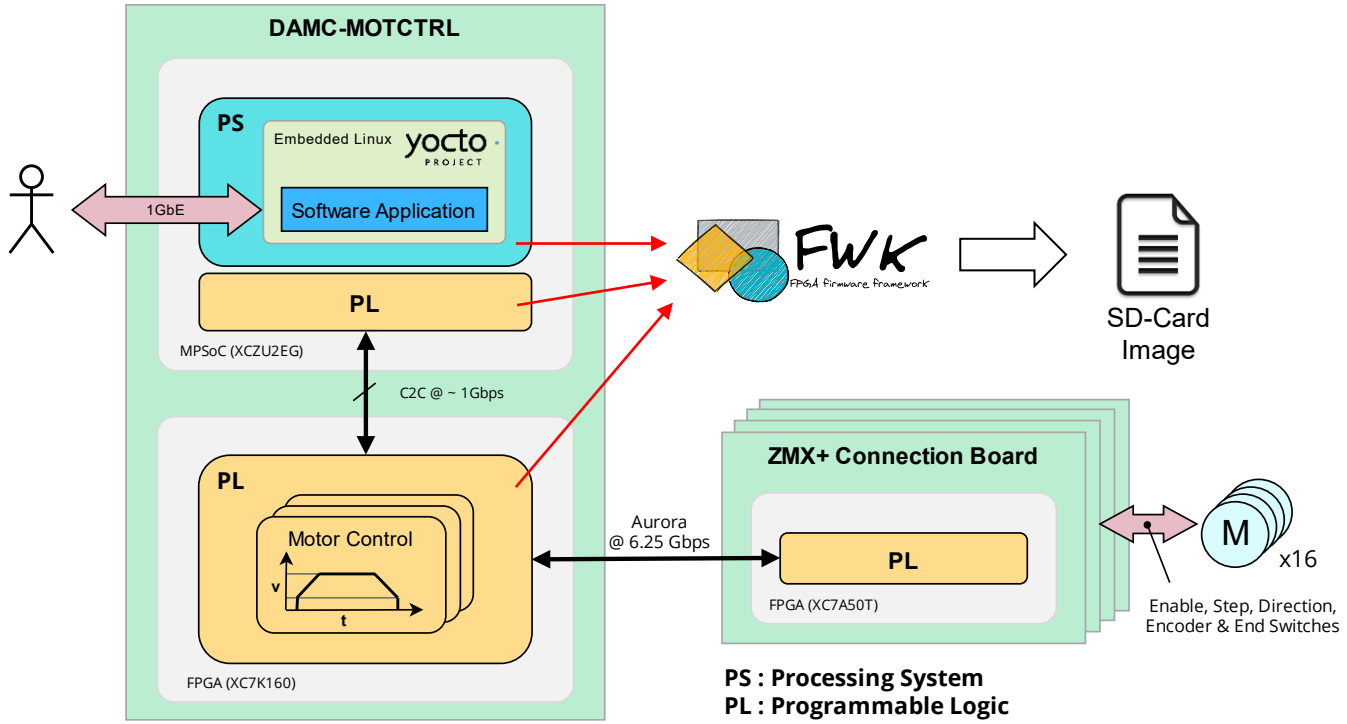
FPGA Firmware Framework (FWK)

Workflow on a Real Project

```
# clone repository
    git clone git@ ... && cd <repo>

# clone all submodules repositories
    git submodules update --init --recursive

# create virtual environment with tools
    make env

# source the Vivado environment
    source /opt/Xilinx/Vivado/2020.2/settings64.sh

# build ZMX+ Connection Board firmware
    make cfg=default_zmx project build

# build DAMC-MOTCTRL (Kintex) firmware
    make cfg=default_k7 project build

# build DAMC-MOTCTRL (Zynq UltraScale+) PL firmware
    make cfg=default_us project build

# build Embedded Linux SD card image
    make cfg=yocto project build
```

The PETRA IV Motion Controller* firmware stack:

- 3 x FPGA designs
- Customized Linux image + boot loader
- C++ application software



*See talk MicroTCA-based Motion Controller for more

Results and Conclusions

Successfully used in MSK group at DESY over 10 years:

- Team of 8 members
- ~40 project
- 15+ distinct hardware boards (many open source)
- Contributions of ~50 developers to the code base

FWK showed remarkable enhancements in:

- Collaboration
- Code quality
- Reproducibility

FWK significantly reduces development time and improves maintainability, but it requires an initial learning curve and ongoing maintenance commitment, which should be carefully considered when implementing the FWK.

Thank you

Check out the source code and documentation:

- **Open-Source FPGA Firmware Framework**

Example designs:

- **SIS8300KU**, **DAMC-FMC2ZUP**, **and many more**

Contact

Deutsches Elektronen-Synchrotron DESY

Cagil Guemues, Burak Dursun, Patrick Huesmann, Michael Randall
MSK

www.desy.de