ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-MODG002

OPEN SOURCE EtherCAT MOTION CONTROL, ECMC, AT PSI

A. Sandström†, A. Acerbo, E. Divall, F. Maier, I. Mohacsi, T. Celcer, X. Wang
Paul Scherrer Institute, Villigen, Switzerland

Abstract

The open-source EtherCAT motion control EPICS module, ECMC, is now being extensively deployed at the Paul Scherrer Institute (PSI), with the number of motion axes currently exceeding 600. ECMC is used across PSI facilities, particularly for the ongoing upgrade of the Swiss Light Source (SLS 2.0), as well as in SwissFEL and HIPA. The scale and diversity of motion control and data acquisition (DAQ) applications have driven the development of many new features in ECMC. Recent developments include support for PVT (Position-Velocity-Time) motion profiles for advanced fly scanning applications, along with beam-synchronous data acquisition capabilities. A concept for flexible and mobile motion control solutions has also been introduced, aiming to support beamline common-pool equipment and facilitate the integration of user-provided motors. Efforts have also focused on improving user-friendliness, defining best practices, streamlining commissioning, and simplifying troubleshooting processes. This contribution highlights the ongoing development and deployment of ECMC within the complex motion control environments at PSI.

INTRODUCTION

The Paul Scherrer Institute (PSI) operates four accelerator facilities: the HIPA and PROSCAN proton facilities, and the SLS and SwissFEL electron facilities. The ongoing upgrade of the Swiss Light Source (SLS 2.0) [1] aims to transform it into a fourth-generation synchrotron by renewing the storage ring and replacing large parts of the facility electronics. For this project, ECMC, an open-source EtherCAT-based motion control module [2], has been adopted as the standard motion control platform. The total scope of the upgrade includes approximately 1500–2000 motion axes, most of them located in the beamlines.

ECMC is fully integrated into the EPICS (Experimental Physics and Industrial Control System) [3] environment, and the preferred way of configuration is by the ecmccfg [4] configuration module. ECMC, maintained and developed at PSI, builds upon the open-source EtherLab EtherCAT master [5] to provide real-time control of EtherCAT [6, 7] devices. Its core functionality includes motion control, soft-PLC capability, and data acquisition, expandable with additional features via a modular plug-in concept.

IMPLEMENTATION ACROSS FACILITIES

Although the SLS 2.0 project is the main driver for ECMC adoption, the framework has also been deployed at SwissFEL and HIPA, where different technical requirements dominate.

SLS

The high number of motion axes and diversity of components have driven efforts to simplify configuration, expand hardware support, and streamline commissioning workflow. Many SLS components require synchronized motion, prompting the development of new features such as Position Velocity Time (PVT) motion. In addition, certain systems are shared between beamlines, leading to the development of a portable motion control concept.

Due to tight schedules and late hardware availability, virtual commissioning techniques have been tested to configure and validate complex systems before physical access.

SwissFEL

While SwissFEL shares many requirements with SLS, its pulsed operation has driven the implementation of beam-synchronous data acquisition features that can capture both motion-related signals as well as other data.

HIPA

At HIPA, applications require interfacing with legacy and radiation-hard components including potentiometers, resolvers, DC motors, and hydraulics that are not typically found in electron accelerators.

SYSTEM ENVIRONMENT

Common across all facilities, ECMC deployment requires a robust, standardized hardware and software environment to ensure uniform operation.

ECMC systems are deployed on standardized hardware at all facilities. Two controller types are used:

- HP DL20 small-form-factor server (preferred), equipped with four additional network ports supporting up to four ECMC IOCs (EtherCAT masters), Fig. 1.
- Beckhoff C6025-0010 industrial PC, a fan less controller (82 × 127 × 47 mm) mountable on a DIN rail, supporting up to two ECMC IOCs, Fig. 2.

All controllers run Debian 12 with a real-time kernel and the PSI EPICS environment. Systems boot diskless via network provisioning using Warewulf [8], which loads the operating system image entirely into RAM at startup. This ensures uniform environments, rapid updates, and improved reliability by eliminating local storage. Real-time thread jitter below 10 µs has been achieved with this platform.

Figure 1: Preferred controller, HP DL20 server.

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-MODG002

Figure 2: Alternative controller, Beckhoff C6025-0010.

ECMC CONFIGURATION WORKFLOW

To accelerate ECMC system deployment, especially for the SLS 2.0 upgrade project, the configuration process has been standardized and partly automated:

- Default values are predefined in configuration files.
- Mandatory scripts are automatically invoked.
- Reusable submodules have been developed for recurring tasks such as synchronization setup.
- A new component library, ecmccomp [9], has been implemented to simplify EtherCAT SDO configuration of mainly motors and encoders

All configurations are defined in an EPICS startup script with the following structure, Fig. 3:

1. ***require ecmccfg***: Load the ECMC configuration framework.
2. ***addSlave.cmd***: Configure communication to an EtherCAT slave.
3. ***applyComponent.cmd***: ecmccomp command to configure device registers (i.e. motor currents, encoder bit count).
4. ***loadYamlAxis.cmd***: Configure motion axis.
5. ***loadYamlEnc.cmd***: Configure additional encoder.
6. ***loadPlcFile.cmd***: Add a soft-PLC logic for synchronization kinematics, control tasks and DAQ.

```
require ecmccfg

# EL5042    2Ch BiSS-C Encoder, RLS-LA11
iocshLoad ${ecmccfg_DIR}addSlave.cmd,      "SLAVE_ID=1, HW_DESC=EL5042"
iocshLoad ${ecmccfg_DIR}applyComponent.cmd "COMP=Encoder-RLS-LA11-26bit-BISS-C,CH_ID=1"
epicsEnvSet(ENC_SID,${ECMC_EC_SLAVE_NUM})

# EL7041    1Ch Stepper
iocshLoad ${ecmccfg_DIR}addSlave.cmd,      "SLAVE_ID=5, HW_DESC=EL7041-0052"
iocshLoad ${ecmccfg_DIR}applyComponent.cmd "COMP=Motor-Generic-2Phase-Stepper, MACROS='I_MAX_MA=1
epicsEnvSet(DRV_SID,${ECMC_EC_SLAVE_NUM})

# Load motion
iocshLoad ${ecmccfg_DIR}loadYamlAxis.cmd,  "FILE=./cfg/axis.yaml, AX_NAME=M1, DRV_SID=${DRV_SID},
iocshLoad ${ecmccfg_DIR}loadYamlEnc.cmd,   "FILE=./cfg/enc_open_loop.yaml, ENC_SID=${DRV_SID}"

# Load PLC code
iocshLoad ${ecmccfg_DIR}loadPLCFile.cmd,   "FILE=./cfg/main.plc, PLC_MACROS='ENC_SID=${ENC_SID},
```

Figure 3: Structure of ECMC startup script.

Hardware Configuration

The hardware configuration is performed in two steps. First, the *addSlave.cmd* script configures communication with each EtherCAT slave. In many cases, additional configuration is required through the slave's Service Data Objects (SDOs) to set device-specific parameters such as motor or encoder settings.

To simplify and standardize this process, ecmccomp has been developed, a component library that automates SDO configuration. Using a single standardized command, *applyComponent.cmd*, ecmccomp applies the correct parameters in the proper units at the appropriate addresses for the selected slave and channel. This abstraction eliminates manual adjustments for device-specific register layouts and units and thereby reduces the risk of misconfiguration.

During axis commissioning, ecmccomp ensures that the configuration workflow remains identical regardless of the hardware in use, which greatly improves consistency, efficiency. At present, ecmccomp primarily supports motor and encoder configuration for a variety of EtherCAT slaves that are used at PSI.

Motion Configuration

Motion axes are managed centrally within an ECMC system i.e. the position loop is centralized but the velocity and current loop is normally executed in EtherCAT drive. Each axis is fully defined in a YAML configuration file containing all relevant parameters, including:

- General: Name, group belonging, unit.
- Trajectory: velocity, acceleration, s-curve, trapetzoidal.
- Encoder: type, scaling, error and warning bits, lookup table, homing parameters.
- Drive: type, scaling, error and warning bits
- Monitoring: limits, maximum velocity, following error.
- Controller: Position loop PID parameters.

The YAML configuration file is then loaded using the command *loadYamlAxis.cmd*. In the default confiuration an EPCIS motor record [10] will also be created for each axis.

Multi encoder support A best practice used at PSI is to configure all relevant encoders for an axis. ECMC supports up to 8 encoders per axis and extra encoders are configured by the command *loadYamlEnc.cmd*.

A typical motion stage at PSI is equipped with a stepper motor combined with a linear absolute encoder as feedback. In such cases, both the linear encoder and the open-loop counter of the stepper motor can be configured.

The primary encoder, used for position control, can be switched at runtime, i.e., in case the linear encoder fails it's possible switching at runtime to control on the open-loop counter to maintain control. Furthermore, during startup, the system can be configured to reference the open-loop incremental counter position with the absolute encoder to ensure a correct initial position.

This configuration adds diagnostic capabilities, by comparing encoder readings over time, it becomes possible to detect issues such as motor stalls or lost steps caused by degrading or non optimal mechanics.

Another use case for multiple encoders is for instance if several encoder sources need to be combined into a single encoder signal. A pratcial example at PSI is the integration of the high accuracy rotary Heidenhain RON905 [11] encoder used at some SLS beamlines. The RON905 encoder consists of 4 high accuracy incremental reading heads offset by 90 degrees. The highest accuracy is achieved by averaging the values from the 4 reading heads. Each reading

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-MODG002

head is configured as an encoder and additionally the motor open loop counter is also added, in total 5 encoders, Fig. 4.

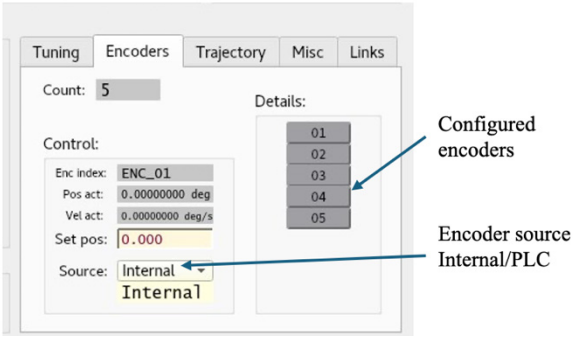

Figure 4: ECMC expert panel, configuration of RON905.

In PLC code the encoders can then be combined into one signal, in this case by calculating the average, Fig. 5.

```
/* The 4 RONs are configured at Enc id 1..4. */
ax${AX_ID=1}.enc.actpos := (mc_get_act_pos(${AX_ID=1},1) +
                            mc_get_act_pos(${AX_ID=1},2) +
                            mc_get_act_pos(${AX_ID=1},3) +
                            mc_get_act_pos(${AX_ID=1},4)) / 4;
```

Figure 5: PLC code for averaging of encoder 1..4.

Several different control modes can be achieved:

- Pure open loop by selecting "Internal" encoder source and selecting the open loop counter as primary.
- Closed loop by using "Internal" source and selecting any of the configured single reading heads as primary.
- Closed loop on the average by selecting "PLC" encoder source, Figure 4.

Synchronized Motion

ECMC has supported synchronized motion since its earliest versions. Originally, kinematics were defined directly in ECMC PLC code using equations [2].

More recently, best-practice configuration templates have been developed for common applications such as slit systems, mirrors, and generic 2–5 DoF systems. In most cases, kinematics are now described using matrices, since this is common and effichient way to define the kinematics, but the approach with equations is sometimes still preferable since it is even more generic.

Once the application-specific kinematic matrices are defined, standard ECMC PLC code snippets can be included to perform the required forward and inverse kinematic calculations, see Fig. 6. The kinematics are then loaded with the *loadPlcFile.cmd* command.

When configuring synchronized motion, it is crucial to manage motion requests across different axes and properly handle edge cases, for example, when a physical axis hits a limit switch, enters an error state, or violates a soft limit.

```
/* Forward kinematics to calculate virtual axes from real axes
   | Tx    |          | Y1 |
   | Yy    |          | Y2 |
   | Pitch | = FWD *  | Y3 |
   | Roll  |          | X1 |
   | Yaw   |          | X2 |
*/
var FWD1[5] := {27/25, -54/25, 27/25,    1/2,    1/2};
var FWD2[5] := {  1/4,    1/2,   1/4,      0,      0};
var FWD3[5] := {   -2,      0,     2,      0,      0};
var FWD4[5] := {    2,     -4,     2,      0,      0};
var FWD5[5] := {    0,      0,     0,     -2,      2};

/* Inverse kinematics to calculate real axes from virtal axes
   | Y1 |          | Tx    |
   | Y2 |          | Ty    |
   | Y3 | = INV *  | Pitch |
   | X1 |          | Roll  |
   | X2 |          | Yaw   |
*/
var INV1[5] := {  0,     1,  -1/4,    1/8,      0};
var INV2[5] := {  0,     1,     0,   -1/8,      0};
var INV3[5] := {  0,     1,   1/4,    1/8,      0};
var INV4[5] := {  1,     0,     0, -27/50,   -1/4};
var INV5[5] := {  1,     0,     0, -27/50,    1/4};

include "axis_kin_mirror.plc_inc"
```

Figure 6: ECMC PLC code defining kinematics in matrices for a 5-axis mirror.

These edge cases are handled by a *master–slave state machine* [12]. Originally implemented in PLC code, this functionality is now built directly into ECMC. The state machine requires explicit identification of which axes act as masters (typically virtual axes) and which act as slaves (typically physical axes). This is achieved by assigning axes to two axis groups in the YAML configuration: one for master axes and one for slave axes. The state machine enforces that only one group can receive motion commands at any given time.

Consider a slit system as an example: the gap and center axes (masters) can move simultaneously, but only after their motion completes the individual blade axes (slaves) can move. If both groups are idle, any axis can accept commands.

The state machine also enforces consistent error handling. If any axis triggers an error such as engaging a limit switch, entering an error state, or receiving a stop command, all axes in both groups are immediately halted, and an error message is generated. An operator must then acknowledge the error and bring the system back into a safe state (inside motion range). For applications requiring more sophisticated edge-case handling, additional logic can be implemented manually in ECMC PLC code. The state machine is added by the command *addMasterSlaveSM.cmd*, Fig. 7.

```
# Master slave state machine
iocshLoad ${ecmccfg_DIR}addMasterSlaveSM.cmd "NAME=test, MST_GRP_NAME=Masters, SLV_GRP_NAME=Slaves"
```

Figure 7: Master Slave State machine.

VIRTUAL COMMISSIONING

For complex motion systems, configurations can be prepared and validated without motion hardware using ECMC virtual axes and PLC code. This approach enables verification of kinematic equations or matrices, interlocks, and other custom functionalities well in advance, significantly reducing pressure during commissioning. This proof of concept approach can also be used to identify potential limitations with mechanical designs, allowing for their correction early in the project. Once the physical hardware

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-MODG002

becomes available, the validated configuration can be applied to the physical axes and all supporting input/output signals; the focus can then shift to tuning the actual motor parameters rather than troubleshooting configuration issues.

An early example of such an approach was used for the pre-commissioning on an in-house designed Front End filter system, Fig. 8, used for beam attenuation with user selectable transmission percentage. The filter system is composed of an upstream water-cooled filter that combined a beam-facing step function with a trailing 6 degrees wedge, and a downstream passively-cooled wedge filter, both on independently retractable positioners with closed-loop encoder feedback. The latter translates at a 6 degree angle. User controls allow for selection of total attenuation thickness in mm and/or transmission percentage based on incoming beam characteristics. Only the water-cooled filter can be exposed to the beam, which is particularly important when changing total attenuation thickness or retracting the filters from the beam.

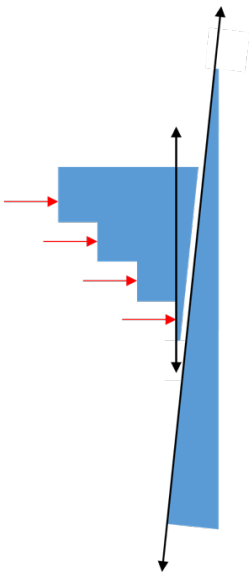

Figure 8: Front End filter system.

The use of virtual axes to mimic the two filter positioners, including off-axis translation of the passively cooled filter, is implemented in an ECMC PLC and emulates closed-loop positioning using the ECMC internal trajectory generator. An additional virtual axis emulating total filter thickness calculates trajectory setpoints for both filter positioners based on total thickness setpoints using a sum of kinematics models of both filters. Inverse kinematics from filter positioner encoder readouts back to total filter thickness provides a closed-loop feedback. One key aspect of this closed-loop axis is that the step function of the upstream filter results in discrete steps of total thickness that are larger than the position lag error. To accommodate this, the position lag error needed to be relaxed. Further, the kinematics model places the upstream filter centered on the step that provides the greatest thickness not exceeding the total filter thickness, and the downstream filter at a position that adds the difference, thereby satisfying the total thickness setpoint. User facing soft limits are implemented on all axes, and are temporarily disabled in the case of total thickness.

PVT MOTION (PROFILE MOVE)

A new feature added to the latest versions of ECMC is support for PVT motion (Position, Velocity, Time). This allows trajectories to be defined using three arrays: position, velocity, and time. The trajectory between two defined points is calculated using a third-order polynomial equation. This way of defining trajectories is convienient for longer time scales, like complete experiments, and therefore complements the ECMC functionality of defining trajetories based on equations which can be convienient for shorter time scales.

PVT motion is configured through the “Profile Move” interface defined in the motor record model 3 driver [10]. This interface allows you to define, build, and execute custom trajectories based on position and time. In this case, the velocity array is derived from the position and time arrays.

Profile trajectories can be applied to any motion axis defined in the ECMC system, including virtual axes. The system also supports output triggers at specified times or positions within the trajectory sequence. For high-accuracy applications, hardware triggers can be assigned to an output defined in the EtherCAT chain. For less demanding applications, software triggers can be implemented through an EPICS process variable.

During execution of a profile trajectory, actual positions and position errors are acquired and latched simultaneously with the output triggers. These data points are buffered in arrays and can be uploaded for analysis during or after the trajectory is completed. This functionality is particularly useful for applications such as fly scanning. One example could be where a certain area needs to be scanned by an opening defined using two slit sets, one horizontal and one vertical, see Fig. 9.

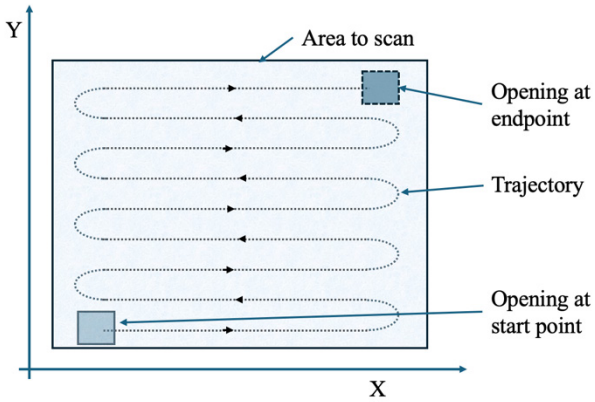

Figure 9: Profile move example: Area snake-scan.

In ECMC, each slit set will be modelled by two virtual axes representing center and gap, and two physical axes controlling the actual blades. The virtual and physical axes are linked by forward and inverse kinematics, Fig. 10.

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-MODG002

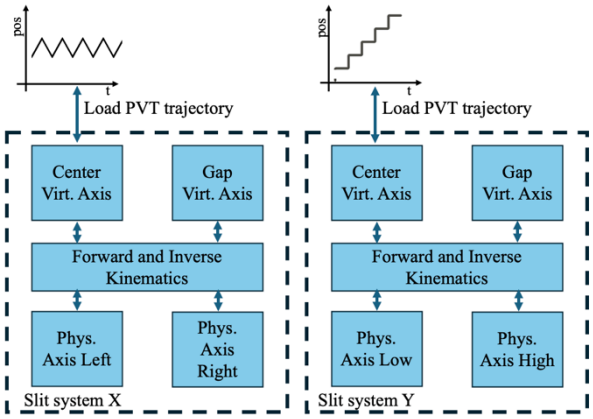

Figure 10: Profile move example, ECMC setup.

By defining a sawtooth trajectory for the horizontal center point axis and a "staircase" trajectory for the vertical center point, the entire area can be scanned. Fig. 11 illustrates the Profile-Move configuration panel.

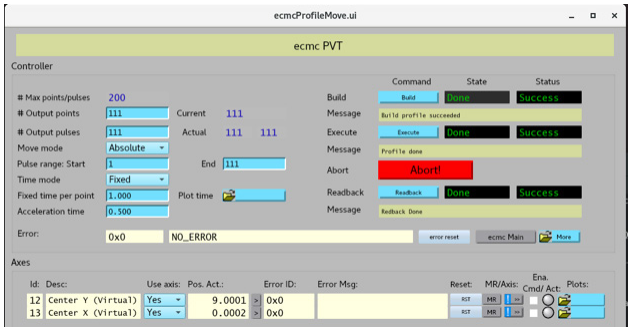

Figure 11: Profile Move Control Panel.

After the move, the resulting actual positions and following error for the entire scan can be read into EPICS PVs, Fig. 12.

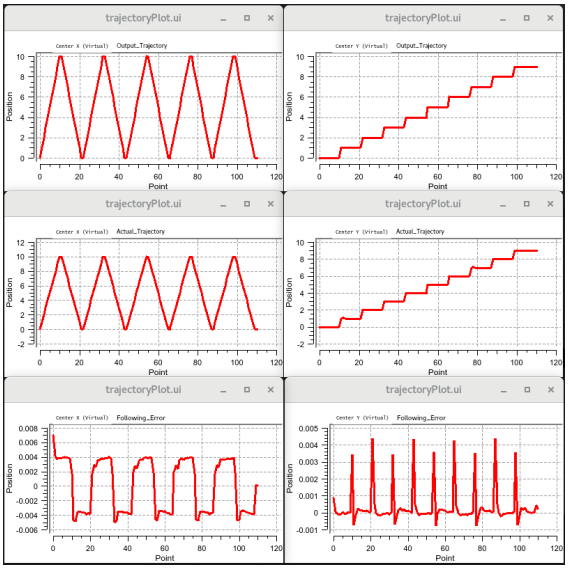

Figure 12: Output trajectory, actual trajectory and following error (left X-axis, right Y-axis).

PORTABLE SYSTEMS

A new concept for portable motion and slow-DAQ systems that can be shared between beamlines has been developed. These systems require only power and a network connection, allowing them to boot and start the ECMC IOC within just a few minutes. Depending on the beamline where the system is deployed, different configurations can be loaded based on either the system's IP address or a hardware ID switch. To achieve this level of flexibility, the compact IPC is used as the hardware platform. This approach provides a lightweight yet powerful solution for applications that need to be moved frequently without complex setup procedures. Figure 13 shows a control box for a portable microscope controlling maximum 4 stepper motors with incremental encoders.

Figure 13: Portable control box.

DAQ

ECMC is also deployed for medium-performance data acquisition, including signals such as motion data, temperatures, load cells, accelerometers, and generic analog or digital inputs. When a slave device or motion axis is configured in ECMC, its data are automatically made available as EPICS process variables (PVs), allowing for straightforward archiving and visualization.

In standard operation, data can be sampled at the EtherCAT cycle rate, typically in the kilohertz range. Certain EtherCAT slaves, however, support higher acquisition rates up to 100 kHz, with data typically collected in 100-element arrays at 1 kHz.

Its also possible to buffer data in ECMC resulting in less frequent but more data for each update.

In applications where frequency-domain analysis is required, an FFT plug-in, *ecmc_plugin_fft* [13], has been developed. This loadable module can compute FFTs for any signal within the ECMC system including motion data like encoder signals, data from EtherCAT slaves, or PLC variables. The plug-in buffers incoming data, performs the FFT, and exposes the results as EPICS arrays. Both continuous and triggered FFT modes are supported, where triggering can be initiated manually via an EPICS PV or through an PLC function call. An example of a spectrum calculated from accelerometer data can be seen in Fig. 14.

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-MODG002

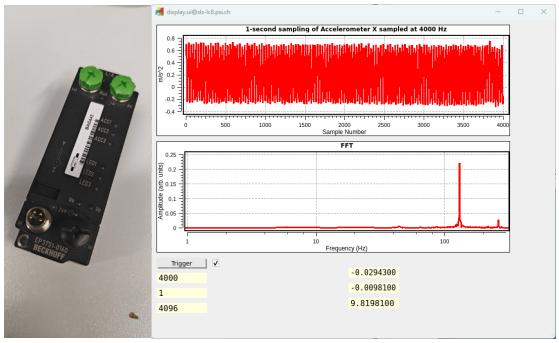

Figure 14: Left: Accelerometer hardware. Right: Raw data and FFT of accelerometer data.

Beam Synchronous DAQ

At SwissFEL, certain data acquisition tasks must be synchronized with the machine rate, i.e. beam-synchronous DAQ. To achieve this, digital triggers from the SwissFEL MRF timing system [14] are connected to an EtherCAT terminal, which timestamps both rising and falling edges using the EtherCAT Distributed Clock (DC). The DC maintains synchronization across the entire EtherCAT network, ensuring that the timing of each machine pulse is precisely known and directly correlated with other data within the ECMC system, Fig. 15. Beam-synchronous operation further requires identification of individual pulses. This is accomplished by varying the trigger pulse length, which can be determined from the latched rise and fall times of the digital input.

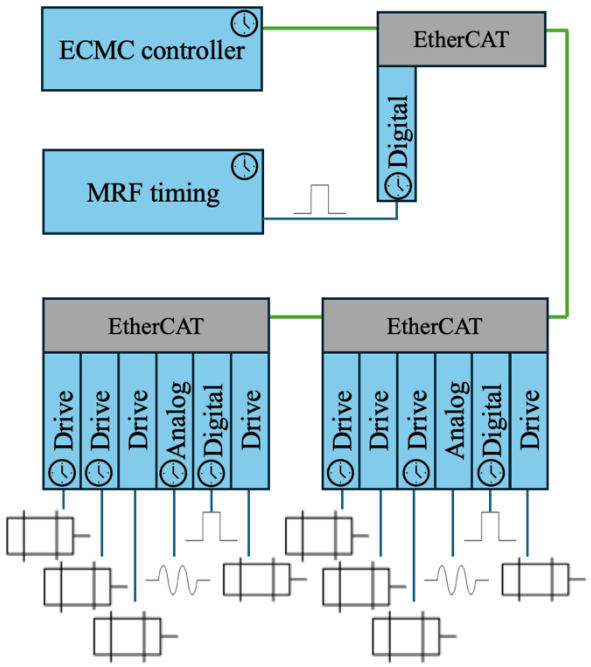

Figure 15: Hardware concept for beam synchronous DAQ.

To simplify the configuration a loadable plugin has been developed, ecmc_plugin_daq [15]. The main functionality of the plugin is that it packs data into an array together with corresponding time stamps (if available) and expose in an EPICS PV as a single array updated each EtherCAT cycle, typically 1 kHz. The advantage with this approach is that all data is acquired in the same EtherCAT cycle. This array can then be further analyzed and the timestamps of the data in the array can be compared and shifted to be aligned with the timestamps of the trigger input. For data originating from EtherCAT slaves with DC timestamps the accuracy can be in the range of 10 microseconds and for slaves without DC-clock, the achievable synchronization performance is in the same range as the EtherCAT cycle time, i.e. millisecond range. Figure 16 shows a panel displaying two analog signals acquired beam synchronous at 20 kHz.

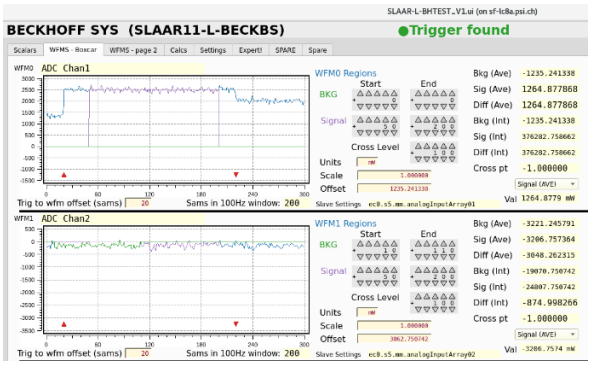

Figure 16: Panel of beam synchronous DAQ. Upper waveform shows signal captured from sample-and-hold detector triggered at 100 Hz.

INTEGRATING RADIATION HARD COMPONENTS

Applications in HIPA require interfacing with radiation-hard components such as analog resolvers and potentiometers. These components are often affected by electrical noise, limited accuracy, and non-linear behavior. Nevertheless, high positioning accuracy is frequently required.

The ECMC platform provides digital filtering capabilities as well as the possibility to apply calibration curves to compensate for these limitations. Resolvers are absolute single-turn devices; however, their accuracy tends to degrade when the rotor aligns with a stator windings, resulting in a cyclic error.

Position accuracy can be significantly improved through calibration. Figure 17 presents repeated measurements of the position error over one full revolution of the resolver, where each dot corresponds to an individual measurement. The repeatability (±0.03°) is considerably better than the overall accuracy (±0.15°). Applying a calibration curve (shown in red) to the encoder object effectively reduces this systematic error and enhances overall system performance.

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-MODG002

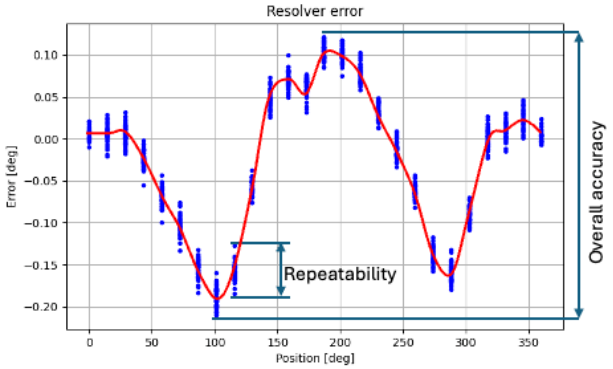

Figure 17: Calibration curve for resolver.

LESSONS LEARNED

In general, the commissioning of most systems proceeded well, but it also provided valuable learnings. The main challenges were related to electrical and mechanical issues, while configuration and software problems were present but less frequent. These experiences highlighted the importance of careful testing and good collaboration during integration. Most of the issues have been documented in the ecmccfg [4] manual to support future commissioning activities.

CONCLUSION

The ECMC motion control framework has matured into a flexible solution deployed across PSI's accelerator facilities. Its open-source nature has been central in enabling rapid adaptation to new hardware, integration of advanced features such as PVT motion and beam-synchronous DAQ, and the development of standardized and efficient commissioning workflows. These qualities have allowed ECMC to scale to control hundreds of motion axes with improved reliability and reduced commissioning effort, while providing a robust and future-proof platform that continues to evolve with community and facility needs.

REFERENCES

[1] T. Celcer, X. Yao, and E. Zimoch, "The SLS 2.0 Beamline Control System Upgrade Strategy", in *Proc. ICALEPCS'23*, Cape Town, South Africa, Oct. 2023, pp. 807-812. doi:10.18429/JACoW-ICALEPCS2023-TUPDP105

[2] T. Gahl *et al.*, "ECMC, the Open Source Motion Control Package for EtherCAT Hardware at the ESS", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, pp. 71-75. doi:10.18429/JACoW-ICALEPCS2017-MOCPL05

[3] EPICS control system, https://epics-controls.org

[4] ECMC confuguration framework ecmccfg, https://github.com/paulscherrerinstitute/ecmccfg

[5] IgH EtherLab, http://www.etherlab.org

[6] EtherCAT Technology Group, http://www.ethercat.org

[7] Beckhoff Automation GmbH, http://www.beckhoff.com

[8] D. Anicic, "Building, Deploying and Provisioning Embedded Operating Systems at PSI", in *Proc. ICALEPCS'23*, Cape Town, South Africa, Oct. 2023, pp.~1505-1507. doi:10.18429/JACoW-ICALEPCS2023-THPDP070

[9] ECMC components library ecmccomp, https://github.com/paulscherrerinstitute/ecmccomp

[10] EPCIS motor record driver, https://github.com/epics-modules/motor

[11] Heidenhain RON905 encoder, https://product.heidenhain.de

[12] A.S. Acerbo, T. Celcer, and A. Sandström, "Open Source EtherCAT Motion Control Rollout for Motion Applications at SLS-2.0 Beamlines", in *Proc. 19th Int. Conf. Accel. Large Exp. Phys. Control Syst. (ICALEPCS'23)*, Cape Town, South Africa, Oct. 2023, pp.~1166-1171. doi:10.18429/JACoW-ICALEPCS2023-TH2BCO02

[13] ECMC Fast Fourier Transform plugin, ecmc_plugin_fft, https://github.com/paulscherrerinstitute/ecmc_plugin_fft

[14] Micro-Research Finland Oy, http://www.mrf.fi

[15] ECMC Data acquisition plugin, ecmc_plugin_daq, https://github.com/paulscherrerinstitute/ecmc_plugin_daq