OPEN SOURCE ETHERCAT MOTION CONTROL ROLLOUT FOR MOTION APPLICATIONS AT SLS-2.0 BEAMLINES

A. S. Acerbo*, T. Celcer, A. E. Sandström, Paul Scherrer Institut, Villigen, Switzerland

Abstract

The SLS-2.0 upgrade project comprises of a new storage ring and magnet lattice and will result in improved emittance and brightness by two orders of magnitude. Paired with these upgrades is a generational upgrade of the motion control system, away from VME based hardware and towards a more modern framework. For SLS-2.0 beamlines, the EtherCAT Motion Control (ECMC) open source framework has been chosen as the de-facto beamline motion control system for simple motion, analog/digital input/output and simple data collection. The ECMC framework comprises of a feature rich implementation of the EtherCAT protocol and supports a broad range of Beckhoff hardware, with the ability to add further EtherCAT devices. ECMC provides soft PLC functionality supported by the C++ Mathematical Expression Toolkit Library (ExprTk), which runs at a fixed frequency on the EtherCAT master at a rate up to the EtherCAT frame rate. This PLC approach allows for implementing complex motion, such as forward and backward kinematics of multi-positioner systems, i.e. roll, yaw, and pitch in a 5-axis mirror system. Additional logic can be loaded in the form of plugins written in C/C++. Further work is ongoing to provide flexible Position Compare functionality at a frequency of 1 kHz coupled with event triggering as a way to provide a basic fly-scan functionality for medium performance applications with the use of standardized SLS-2.0 beamline hardware. We provide an overview of these and related ECMC activities currently ongoing for the SLS-2.0 project.

ETHERCAT MOTION CONTROL ARCHITECTURE

The EtherCAT open standard has firmly established itself as a reliable real-time fieldbus for extensively distributed and synchronized systems [1]. Diamond Light Source (DLS) [2], European Spallation Source (ESS) [3, 4], SPring-8-II [5], Paul Scherrer Institut (PSI), and others have introduced open-source solutions for the bus master in scientific installations, utilizing EtherCAT hardware for digital and analog I/O and motion control. At these institutes, EtherCAT serves mid-performance data acquisition and motion control in accelerator and beamlines applications.

The open source EtherCAT Motion Control (ECMC) [6] framework originally developed at the ESS uses the open source Etherlab master [7] to control the EtherCAT bus. ECMC interfaces with EPICS Motor Record using a model 3 driver and implements basic functionality such as positioning, homing, and limits. The ECMC framework supports advanced features including full servoloop feedback, virtual axes, interlocking, and a PLC-like scripting language based on the C++ Mathematical Expression Toolkit Library (ExprTk) [8].

ECMC Configuration and EPICS IOC

Configuration of EPICS IOCs using the ECMC framework is facilitated by the ECMC configuration (ECMCCFG) framework [9]: The configuration framework includes all the essential files required for setting up an EPICS IOC for EtherCAT-based motion control and DAQ. For a given EtherCAT master, individual EtherCAT slaves can be added with the addSlave script and specifiying the exact model of EtherCAT module to be added. For the addSlave script, the following parameters are accepted:

- HW_DESC: Hardware descriptor, i.e. EL1008
- SLAVE_ID: (optional) bus position
- SUBST_FILE: (optional) substitution file
- P_SCRIPT (optional) naming convention prefix script
- NELM: (optional) Used for oversampling cards. Defaults to 1
- DEFAULT_SUBS (optional) option to disable default PVs for mapped PDOs

In the above function, the HW_DESC parameter loads EtherCAT specific parameters. Further scripts are available for adding custom (local) configuration parameters to specific slaves, modifying SDO objects, enabling diagnostics, or loading plugins from PLC. Individual axes are typically added with the addAxisYaml script, which accepts a yaml file containing configuration parameters for a single axis. Yaml configuration files are parsed with jinja2; Yaml fields for axes include:

- axis: id, mode (CSV, CSP), power on/off parameters
- var: local variables
- epics: EPICS related fields, such as PV name, EGU, PREC
- drive: motor driver parameters
- encoder: motor encoder parameters
- controller: PID loop tuning parameters
- trajectory: trajectory generation
- input: limits, homing switch, interlocks
- plc: PLC related fields
- homing: homing routine related
- softlimits: enable/disable and limit values
- monitoring: limits on velocity, tracking lag, setpoint deadband

For applications requiring highly customized axes where functionality provided by ECMC PLC is not sufficient, a plugins allowing external functions written in C/C++ to be loaded and called from PLC code. All ECMC variables are available in real time and can be passed to these external functions. For data acquisition and processing, local storage

* alvin.acerbo@psi.ch

buffers can be created to accumulate data across EtherCAT frames, or for oversampling purposes. These buffers can then be read out by external functions or linked to EPICS records. This is of particular interest at higher EtherCAT frame rates that exceed EPICS scan rate.

ECMC PLC Syntax

The PLC control offered by ECMC provides direct control over select EtherCAT data, motion variables (axis trajectory generation, encoder readout, drive power, limits, etc.), PLC local and global variables, data storage objects, and custom functions loaded via plugin. The basic building blocks can be used to easily set up for example axis slaving, synchronization, interlocks that are evaluated at EtherCAT frame rate:

- slaving: setPos2:=actPos1;
- synchronization: setPos2:=setPos1;
- phasing: setPos3:=setPos1+setPos2;

Complexity can be added with the supported ExprTk toolkit to implement simple motion kinematics, implement state machines, or implement Position Compare functionality as part of a fly-scanning configuration.

ECMC APPLICATIONS AT SLS-2.0 BEAMLINES

ECMC has been adopted at several beamlines at the Swiss Light Source (SLS) in a limited fashion alongside existing controls infrastructure [10]. With the current upgrade of the SLS, the controls infrastructure of the accelerator and beamlines are also being upgraded. For beamline motion applications, this includes nearly all motion devices that interact with beam transport into the experimental hutch, as well as a large share of motion devices in the experimental hutch. Early applications include slit systems and multi-axis mirror systems with high precision kinematics.

XY Slit System

The most commonly used XY slit and diaphragm systems at SLS-2.0 are composed of 2 L-shaped or four individual blades driven by servo motors, and define the center position and gap size of the delimited beam in horizontal and vertical axes. Each servo motor is controlled by a Beckhoff EL7211 terminal, which integrates motor drive, encoder readback, and limit switches (Fig. 1). An EPICS IOC using the ECMC Motion Control Module for EPICS as well as the configuration framework for ECMC is used to set up CSV motion control over the four physical positioners. At the ECMC level, low and high travel soft-limits are imposed and PID parameters are defined. The physical positioners are then controllable via EPICS PVs: position setpoint, position readback, instantaneous velocity, limit activation, etc, are all directly served to the EPICS IOC. Optionally, a level 3 Motor Record can be enabled to provide compatibility with facility procedures.

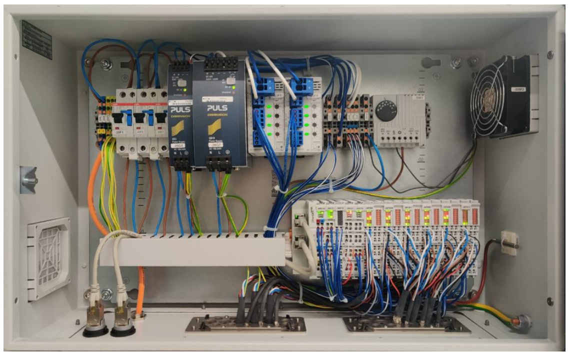
Figure 1: SLS-2.0 EtherCAT enclosure with power supply and fuses (top) and Beckhoff modules (bottom right) for a 4-Axes XY slit system and a 2-Axes diagnostic screen.

Control over the slit center and gap in horizontal and vertical axes is accomplished by a set of 4 individual virtual positioners: 2 for each axis. Similar to the ECMC implementation of physical positioners, a virtual positioner has common motion elements, such as a setpoint, a readback link, limits, and a trajectory generator. In the case of a virtual positioner controlling slit center position, PLC level logic can be used to implement inverse kinematics using the real-time positions of the physical positioners as follows:

$$center = \frac{leftBlade + rightBlade}{2} , \tag{1}$$

$$gap = leftBlade - rightBlade . \tag{2}$$

Further, forward kinematics can be implemented to calculate setpoints for the physical positioners:

$$leftBlade = \frac{center - gap}{2} , \tag{3}$$

$$rightBlade = \frac{center + gap}{2} . \tag{4}$$

Although simultaneous motion of the physical positioners is not impeded by the addition of these virtual positioners, extra effort must be made to allow for simultaneous motion of multiple virtual positioners that control a single set of physical positioners: A drift of gap size may result from a change in center position unless the setpoints of all virtual positioners are controlled. If this is done correctly, simultaneous motion of gap and center using the default trajectory generator and standard kinematics equations, is possible.

Control over physical positioners by virtual positioners switches the internal trajectory generator of the physical positioner off, such that an external driver (i.e. output of kinematics equations of a virtual positioner) can drive the physical positioner instead. This works well, but prevents simultaneous control of a physical axis by both the physical positioner and a virtual positioner. A simple state machine can be implemented at the PLC level of each virtual axis to govern the internal setpoint of a physical axis to be directed by the internal trajectory generator or external driver (Fig. 2). This state machine ensures that the activation of a virtual positioner (S0→S1) places all affected physical positioners into external drive mode. While the virtual axes are enabled (S1→S1), inverse kinematics can then be used to direct the physical positioner(s). A transition S1→S0 is made when the virtual positioner has achieved its setpoint: the physical positioners are then placed back into internal trajectory mode and can be controlled directly.

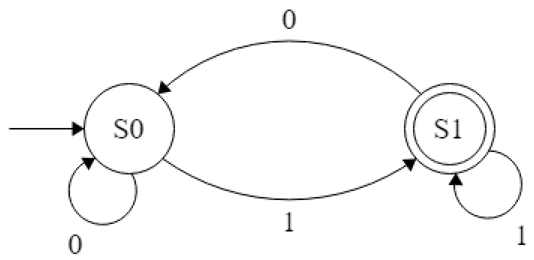

Figure 2: A typical virtual axis state machine with active (S0) and inactive (S1) states. This state machine can be implemented at the PLC level, where each state transition can be tied to execution of a particular block of code.

5-Axes Mirror System

The first two sets of beamline mirror systems commissioned at SLS-2.0 were the focusing and collimating mirrors for the newly built Debye beamline. The motion of the mirror optical surface is controlled by 5-Axes: Independent X translation stages on the upstream and downstream ends of the mirror, and independent Y translation stages on the upstream inboard end, the upstream outboard end, and the downstream end. Beckhoff EL7041 and Beckhoff EL5042 terminals were used for motion control and encoder feedback. Forward and inverse kinematics were implemented similarly to the XY slit system: PLC level logic implements the forward and inverse kinematics equations given the relative positions of the axes in the XY plane. For a 5-axes system with 2 X translational axes and 3 Y translational axes, kinematics equations can be parameterized to use several physical dimensions of the mirror mechanics and several variable offsets that can be used for stripe selection in the X plane and height offset in the Y plane. For inverse kinematics, planar rotation angles pitch θ, yaw ψ, and roll ϕ are readily obtained directly from the 5 physical mirror positioners

$$\theta = \arctan \frac{\frac{Y_{UI}+Y_{UO}}{2} - y_D}{dist_Y} , \tag{5}$$

$$\psi = \arctan \frac{X_U - X}{dist_Y} , \tag{6}$$

$$\phi = \arctan \frac{Y_{DI} - Y_{DO}}{dist_X} , \tag{7}$$

where X$_D$ and X$_U$ refer to downstream and upstream X translational stages, Y$_D$ to downstream Y translational stage, and Y$_{UI}$ and Y$_{UO}$ to upstream inboard and upboard Y translational stages, dist$_Y$ to the distance between the Y$_D$ and Y$_{UI}$ in the Y plane, and dist$_X$ is the distance between X$_U$ and X$_D$ in the X plane. From there, center points of the mirror system in the X, Y, and Z planes can be calculated as follows:

$$X_{CP} = \bar{X} \cos\phi - (I3 + Y_{OFF}) \sin\phi + X_{OFF} \cos\phi \cos\psi + A1 , \tag{8}$$

$$Y_{CP} = X_{OFF}(\sin\theta \sin\psi + \cos\theta \cos\psi \sin\phi) - (y_{gcs} - Y_{US}) - D6 \sin\theta + (I3 + Y_{OFF}) \cos\theta \cos\phi , \tag{9}$$

$$Z_{CP} = D1 - X_{OFF}(\cos\theta \sin\psi - \cos\psi \sin\theta \sin\phi) + D6 \cos\theta + (I3 + Y_{OFF}) \cos\phi \sin\theta + \bar{X} \sin\theta \sin\phi , \tag{10}$$

where A1 is the shift in X from center of beam to center of ball of Y$_D$, I3 is the distance from ball of Y$_D$ to mirror surface, Y$_{OFF}$ and X$_{OFF}$ are the offset corrections between mirror stripe optical surfaces in the X and Y plane, and D6 is the shift in Z from center of ball of Y$_D$ to center of mirror.

The approach for forward kinematics uses two helper functions to avoid redundant calculations:

$$A6 = A1 - X_{CP} - (I3 + Y_{OFF}) \sin\phi + X_{OFF} \cos\phi \cos\psi , \tag{11}$$

$$A3 = X_{CP} \cos\theta \sin\phi - A1 \cos\theta \sin\phi - D6 \cos\phi \sin\theta - Y_{CP} \cos\phi + (I3 + Y_{OFF}) \cos\theta \cos(\phi)^2 + (I3 + Y_{OFF}) * \cos\theta \sin(\phi)^2 + X_{OFF} \cos\phi \sin\theta \sin\phi . \tag{12}$$

From there, forward kinematics equations for each physical axes are as follows, where I1 refers to the distance between Y$_{UI}$ and Y$_{UO}$, and I2 to the distance in the Z plane between Y$_D$ and Y$_U$:

$$X_U = -\frac{A6}{\cos\phi} - \frac{I2 \tan\psi}{2} , \tag{13}$$

$$X_D = -\frac{A6}{\cos\phi} + \frac{I2 \tan\psi}{2} , \tag{14}$$

$$Y_D = Y_{gcs} - \frac{A3}{\cos\phi} , \tag{15}$$

$$Y_{UI} = I2 \tan\theta + \frac{Y_D - I1 \tan\theta}{2} , \tag{16}$$

$$Y_{UO} = I2 \tan\theta + \frac{Y_D + I1 \tan\theta}{2} . \tag{17}$$

The above kinematics equations are evaluated at the PLC level at a default ECMC rate of 1 kHz on a dedicated core of an Intel(R) Xeon(R) E-2278G CPU @ 3.40 GHz. All virtual positioners are controllable simultaneously; since the trajectories of all physical axes are evaluated in PLC realtime, the individual axes setpoints accurately follow kinematics described trajectories, and are predominantly subject to the stability of PID loop tuning and mechanical design limitations. Care must be taken that the velocity and acceleration/deceleration of the virtual positioners are chosen to not result in trajectories of underlying physical positioners that exceed their own velocity and acceleration/deceleration limits.

Position Compare

In a traditional step scanning mode of motion axes, the axes move in a stepwise fashion, at each target position coming to a complete halt and yielding to other processes to start and stop before moving on to a next target. In the case of a hardware defined step-scan and detector trigger-chain implementation, the system overhead is largely a result of the deceleration and acceleration of the motion axis, their settling time, and the duration of the trigger generating event. These factors are independent of detector dwell time. In a fly scanning modality, the most common implementation treats the motion control system as the master, which directs one of more axes to move in a continuous fashion along the entire trajectory profile, while detector(s) (the slaves) are triggered in a synchronized or unsynchronized fashion based on the trajectory profile. Typically, real-time readout of the motion encoder serves as a basis for this synchronization event. In an alternative implementation, one or more detectors can serve as the master and dictate the continuous motion of one or more motion axes.

A position compare loop can be implemented in an ECMC PLC, where the current encoder position is compared with a defined set of target positions at a default ECMC rate of 1 kHz. Target positions can be pre-computed coordinates along the trajectory axes, computed on-the-fly based on external inputs, or be defined as relative targets on a position or time axis. EPICS channel access is used to expose control over calculations of these position compare trajectory targets, which includes total trajectory distance, a start point or trigger signal, and segment distance or duration. Trigger output options from the position compare loop are ideally suited to enable output signals on digital or analog output capable EtherCAT modules, which are then active on the next EtherCAT frame. Although the position compare loop is evaluated at 1 kHz, output signals can be oversampled using appropriate modules (i.e. EL4732) to increase temporal resolution of the trigger signal beyond the EtherCAT bus cycle time. Optionally, timestamping digital output modules can be used (i.e. EL2252) that switch their outputs to match transferred timestamps. Oversampling of encoder input is technically feasibly, but requires well-defined trajectories, well-tuned axes, and forward interpolation logic as part of the position compare PLC logic.

To test basic position compare functionality in an ECMC PLC, a pair of servo-drive linear stages (Beckhoff AM8112-0FH0-0000) were controlled by Beckhoff EP7211-0034 modules, which incorporates motor drive, encoder readout, and limits in a single module. A Beckhoff EL2808 digital out was used to send triggers via digital output and illuminate LEDs on the module itself for visual confirmation. For the implementation of an ECMC PLC with position compare logic, an approach similar to that of the "PCOMP" Position Compare block in the PandABox was chosen [11], where settable parameters are exposed to the user via EPICS channel access. Settable parameters include position compare parameters, trigger generation parameters, position compare trajectory targets, and output trigger links. Several status and diagnostic parameters are available as well.

The two translational stages were used to create a logarithmic spiral [12] scanning virtual axis with polar equation:

$$r = ae^{b\theta} \ . \tag{18}$$

Forward kinematics of virtual axes for the polar angle and radius of a logarithmic spiral arch were implemented according to standard Cartesian to polar coordinate conversion equations:

$$r = \sqrt{x^2 + y^2} \ , \tag{19}$$

$$\theta = -\arctan2(y, x) - \pi/2 \ . \tag{20}$$

Further, forward kinematic of a virtual axis representing the positive logarithmic spiral arch was created, with the spiral origin (1,0) defined as starting position and incrementing with increasing arc length. We opted for the closed form describing the arc length from the spiral origin as:

$$L(\theta) = \frac{a\sqrt{1+b^2}e^{b\theta}}{b} \ . \tag{21}$$

Evaluation of the kinematics was done in an ECMC PLC at a 1 kHz frame rate using the ExprTk functionality in ECMC. Optionally, for more complex equations, this can be offloaded to a C/C++ program using external libraries, for example the GNU Scientific Library [13]. Those functions can then be loaded as an ECMC plugin and called from an ECMC PLC. Inverse kinematics to define X and Y position based on θ was done using:

$$x(\theta) = a\cos\theta e^{b\theta} \ , \tag{22}$$

$$y(\theta) = a\sin\theta e^{b\theta} \ . \tag{23}$$

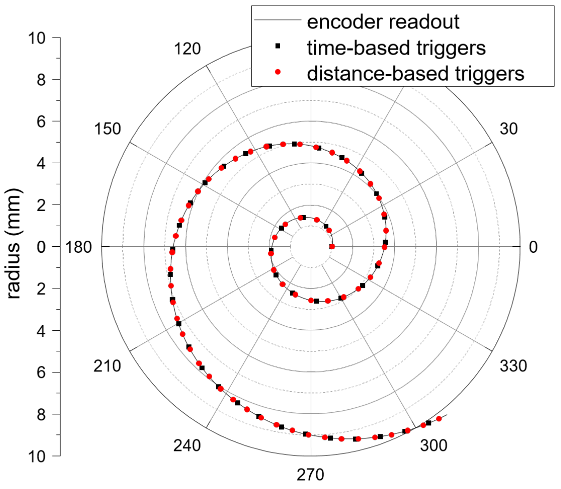

Figure 3: Polar plot of a logarithmic spiral as $r = ae^{b\theta}$, with a=1, b=0.2. Polar coordinates r and θ were calculated with forward kinematics implemented in an ECMC PLC. Location of trigger signals from time based (black square) and distance based (red dot) position compare are overlaid.

Position compare logic was used along the spiral arc virtual axis. As a proof of concept, position compare with time and distance based intervals was used to generate triggers on the closed-loop spiral arc axis in motion at a constant velocity of 1.0 mm/s as shown in Fig. 3. Here, time based triggering was set at 1.2 s intervals and distance based triggering at 0.8 mm intervals, settings which were chosen to properly display triggering data points over a longer trajectory. This corresponds to trigger signals approximately every 1200 and 800 EtherCAT frames for the time-based and distance-based options, respectively.

Constant linear motion along the logarithmic spiral trajectory after an initial ramp up is plotted in Fig. 4, where trigger points are overlaid. Both trigger options show equidistant spaced intervals in the contant velocity domain of the virtual axis, from t=2 to t=50.

Customizable triggering can be implemented to meet the needs to trigger-receiving electronics: If supported by the appropriate EtherCAT output modules, trigger width and offset defined by ECMC PLC can meet a wide range of needs. Further, with analog out triggering, voltage levels can easily be defined.

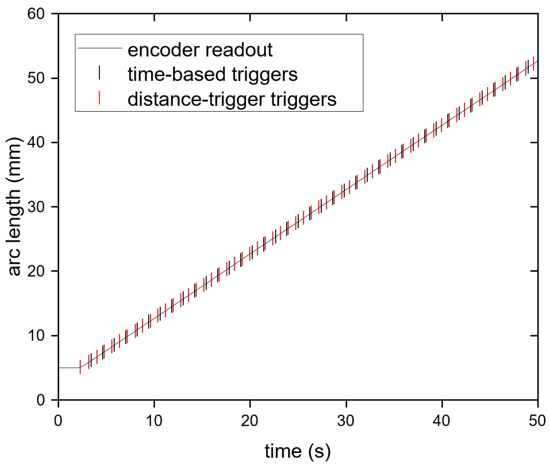

Figure 4: Time-trajectory of the logarithmic spiral in Fig. 3 showing a linear progression after in initial ramp-up betweent t=0 and t=2. Trigger locations from time based (black mark) and distance based (red mark) are overlaid.

CURRENT PROGRESS AT PSI

Besides general deployment of ECMC at SLS-2.0 beamlines, girders, and RF systems, several topics are under active development at PSI.

Integration of Fly-Scanning in a Beamline Experimental Control Layer

A Beamline Experimental Control layer based on BlueSky [14] is under active development at SLS-2.0 beamlines [15]. A hardware abstraction layer based on Ophyd integrates analog/digital signals and axes via standard Ophyd objects and more complex instrumentation by custom Ophyd objects. Early work towards a hardware agnostic fly-scanning interface is being trialed with PandABox [11] and ECMC-based hardware driven fly-scanning configurations.

Requirements for fly-scanning at SLS-2.0 were collected and categorized into 3 performance tiers (Table 1). These requirements were then matched with several fly-scanning solutions and strategies that are either currently available or under active development at PSI. Figures for Cost are per axis and capture only the cost of hardware related to deployment, and not development. Encoder sampling rate and the rate of the PC logic severely affect application potential, and are therefor typically the most critical hardware aspect of a fly-scanning solution. The qualitative comparison for flexibility aims to capture the potential to address the technical requirements expecting for performance: whereas a software-based approach offers broader access to signals and facility resources, a purely hardware based solution typically has access to a limited set of resources. The qualitative comparison for deployment aims to capture the ease of deployment and takes into account required facility resources to deploy and maintain such system.

Table 1: Fly-Scanning Performance Tiers

	Tier A	**Tier B**	**Tier C**
Type	Software	ECMC	SmarAct
Axis cost (CHF)	< 100	100-1000	> 1000
Encoder sampling	≤ 10 Hz	1 - 100 kHz	> 10 kHz
PC loop	≤ 10 Hz	1 - 10 kHz	> 10 kHz
Flexibility	++	++	+
Deployment	+++	++	+
% Use at SLS-2	~ 20	~ 75	~ 30

For Tier A, a pure software-based approach offers a low cost option as it uses existing hardware. PC loop logic is implemented in the BEC layer, and is thus subject to network latency and is suitable for low-performance applications. An ECMC-based solution using standard SLS-2 ECMC hardware can offer guaranteed performance and a higher performance tier as it implements PC logic at the PLC level tied to the EtherCAT frame rate. For high performance applications, specialty dedicated controllers such as Newport, ACS, or SmarAct are recommended within the SLS.

High Performance ECMC Applications

For high performance applications requiring fast reaction times, tuning of the controller and ECMC server real-time environment is important. Tuning parameters include:

- Thread affinity
- Thread priorities
- Isolation of cores
- EtherCAT driver (native or generic)
- Power saving settings

Recent tests show that running multiple ECMC systems on the same controller at a 10 kHz sample rate is feasible, even with motion axes. As an example, two identical motion systems were executed on one ECMC server with two EtherCAT masters:

- 10 EtherCAT slaves each per master
- 8 motion axes, 4 physical and 4 virtual, each
- 10 kHz sample rate
- ECMC RT thread affinity on dedicated cores
- Native EtherCAT driver (ec_igb)

The maximum jitter of the real-time threads was measured to be below 6 µs which results in a good margin to the 100 µs scan rate.

Synchronization of Insertion Device and Beamline Optics

Isolation of EtherCAT networks by domain, beamline segment, or experimental setup eases management and reduces down time. However, for some applications, connectivity between EtherCAT networks is desired. Communication between EtherCAT networks is made possible by an EtherCAT bridge terminal (i.e. EL6692), which serves to synchronize distributed clocks and enable real-time data exchange between isolated EtherCAT networks. This functionality is of particular interest to applications that require a high degree of synchronization of two EtherCAT networks, such as synchronizing an undulator gap with monochromator energy axis during spectroscopy experiments. Work is ongoing towards on flexible approach to linking to EtherCAT networks, and defining how to resolve differences in timebases.

CONCLUSION

The adoption of ECMC as the standard control system for motion control and analog/digital I/O for beamlines at SLS-2.0 is currently underway. Successful adoption of ECMC as controls system for several beamline components at select SLS beamlines in the past few years has paved the way for a near-complete replacement of the aging VME-based control system with the more modern and industry supported alternative. The flexible and modular approach to adding individual EtherCAT modules, and thus offering a highly customizable and tailored configuration for individual beamline components is a significant improvement. Further, the ability to monitor and act on EtherCAT signals at a guaranteed and moderately high rate makes ECMC well suited to handle a variety of more complex beamline tasks, such as implementing virtual motors, offering medium performance data acquisition, and allowing flexible fly-scanning strategies.

ACKNOWLEDGEMENTS

The authors would like to thank all contributors to the Open Source EtherCAT project, colleagues at the PSI Contols Systems section for their valuable input and hardware support, Niko Kivel for assisting in early adoption of ECMC, and Stephan Hitz for providing kinematics equations for the 5-Axes Mirror System.

REFERENCES

[1] EtherCAT Technology Group, https://www.ethercat.org/en/technology.html

[2] R. Mercado, J. Rowland, I. Gillingham, and K. Wilkinson, “Integrating EtherCAT based I/O into EPICS at Diamond”, in *Proc. ICALEPCS’11*, Grenoble, France, Oct. 2011, pp. 662–665. paper WEMAU004

[3] T. Gahl, D. Brodrick, T. Bögershausen, O. Kirstein, T. Korhonen, D. Piso, and A. Sandström, “ECMC, the Open Source Motion Control Package for EtherCAT Hardware at the ESS”, in *Proc. ICALEPCS’17*, Barcelona, Spain, Oct. 2013, pp. 71–75. doi:10.18429/JACoW-ICALEPCS2017-MOCPL05

[4] J. Etxeberria, J. H. Lee, and A. Sandstrom, “EtherCAT Open Source Solution at ESS”, in *Proc. ICALEPCS’19*, New York, USA, Oct. 2019, pp. 1195–1198. doi:10.18429/JACoW-ICALEPCS2019-WEPHA046

[5] M. Ishii, M. T. Takeuchi, C. Kondo, and T. Fukui, “A Control Syustem using EtherCAT Technology for the Next-Generation Accelerator”, *Proc. ICALEPCS’19*, New York, USA, Oct. 2019, pp. 1258–1261. doi:10.18429/JACoW-ICALEPCS2019-WEPHA068

[6] ECMC Motion Control Module for EPICS, https://github.com/epics-modules/ecmc

[7] IgH EtherCAT Master for Linux, https://gitlab.com/etherlab.org/ethercat

[8] C++ Mathematical Expression Toolkit Library (ExprTk), www.partow.net/programming/exprtk

[9] Configuration scripts for EtherCATMotion Controller (ECMC), https://github.com/paulscherrerinstitute/ecmccfg

[10] T. Celcer, E. Zimoch, and X. Yao, “The SLS 2.0 Beamline Control System Upgrade Strategy”, presented at ICALEPCS’23, Cape Town, South Africa, Oct. 2023, paper TUPDP105, this conference.

[11] S. Zhang, Y.M. Abiven, J. Bisou, G. Renaud, G. Thibaux, F. Ta, S. Minolli, F. Langlois, M. Abbott, T. Cobb, C.J. Turner, and I.S. Uzun, “PandABox: A Multipurpose Platform for Multi-Technique Scanning and Feedback Applications”, in *Proc. ICALEPCS’17*, Barcelona, Spain, Oct. 2017, pp.143–150. doi:10.18429/JACoW-ICALEPCS2017-TUAPL05

[12] C. Baumgarten and G. Farin, “Approximation of logarithmic spirals”, *Comput. Aided Geom. Des.*, vol. 14, no. 6, pp. 515-532, 1997. doi:10.1016/S0167-8396(96)00043-X

[13] M. Galassi *et al.*, *GNU Scientific Library Reference Manual*. Network Theory Ltd., 2009.

[14] D. Allan, T. Caswell, S Campbell, and M Rakitin, “Bluesky’s Ahead: A Multi-Facility Collaboration for an a la Carte Software Project for Data Acquisition and Management”, *Synchrotron Radiat. News*, vol. 32, no. 3, pp. 19–22, 2019. doi:10.1080/08940886.2019.1608121

[15] K. Wakonig, C. Appel, A. Ashton, S. Augustin, M. Holler, I. Usov, J. Wyzula, and X. Yao, “A Beamline and Experiment Control System for SLS 2.0”, presented at ICALEPCS’23, Cape Town, South Africa, Oct. 2023, paper MO2AO02, this conference.