ON-THE-FLY SCANS FOR FAST TOMOGRAPHY AT LNLS IMAGING BEAMLINE

G. B. Z. L. Moreno[1], F. P. O'Dowd, H. H. Slepicka, R. Bongers, M. B. Cardoso
CNPEM - LNLS, Caixa Postal 6192, Campinas – 13083-970, Brazil

Abstract

As we go to brighter light sources and time resolved experiments, different approaches for executing faster scans in synchrotrons are an ever-present need. In many light sources, performing scans through a sequence of hardware triggers is the most commonly used method for synchronizing instruments and motors. Thus, in order to provide a sufficiently flexible and robust solution, the X-Ray Imaging Beamline (IMX) at the Brazilian Synchrotron Light Source [1] upgraded its scanning system to a NI PXI chassis interfacing with Galil motion controllers and EPICS environment. It currently executes point-to-point and on-the-fly scans controlled by hardware signals, fully integrated with the beamline control system under EPICS channel access protocol. Some approaches can use CS-Studio screens and automated Python scripts to create a user-friendly interface. All programming languages used in the project are easy to use and to learn, which allows high maintainability for the system delivered. The use of LNLS Hyppie platform [2, 3] also enables software modularity for better compatibility and scalability over different experimental setups and even different beamlines.

INTRODUCTION

The traditional methods for performing scans and controlling experiments at the beamline can present deadtime issues even when comparing the performance of point-to-point scans. Workarounds to the most common causes, like network latency or inefficient programming, may include the control of part of the experiment solely by hardware triggers, orchestrated by low-level implementations. At LNLS, off-the-shelf FPGA-based systems present a standard solution to controlling experiments and integrating equipment at the beamline. EPICS channel access transfers information between low-level device drivers and high-level user interfaces in the form of *EPICS Process Variables*, or EPICS PV's.

The communication between the top and bottom layers of this system architecture takes place over the network in TCP-IP protocol and thus, is non-deterministic. For fast-scan applications, one cannot disregard network latency, inherent to the communication between motion controllers, detectors, and other beamline instruments such as fast shutters and counters triggered from EPICS.

At the IMX Beamline, white beam photon density is of the order of 10^{14} ph/s/mm^2, with an energy range of 4 to 25 KeV. Hence, typical exposure times can vary from 50 to 300 ms for the most common samples. Adding this to the camera's 468 ms readout time at full resolution, frame period could reach more than 500 ms. Due to network overhead while interpreting EPICS PVs, frame period can reach almost a second in a conventional scan. The hardware synchronization is a way to eliminate this latency given the fact that, after its start, the scan runs independently under low-level applications, not relying on communication over the network.

SYSTEM DESCRIPTION AND DATA ACQUISITION ARCHITECTURE

The LNLS X-Ray Imaging Beamline (IMX) uses a National Instruments PXI-6602 timing board attached to an NI PXI-1045 chassis to read digital counters and trigger devices during scans. A Huber 409 rotation stage performs the sample rotation during tomography. The scan points are programmed directly on Galil DMC-4183 motion controllers as a function of parameters passed from EPICS PV's to reduce latency between scan points and enable quick tomography. The controller sends feedback through the I/O interface on trippoint arrival and transmits the present motion status, enabling hardware synchronization between motors and scan devices. A LabVIEW VI running on a dedicated Windows Machine controls a PCO.2000 camera, which is currently the main detector at IMX Beamline. To prevent high x-ray dose on the sample, a Uniblitz XRS25 shutter blocks the x-ray outside acquisition interval. The user interface screens run from a workstation on CS-Studio environment under Linux Red Hat operational system. CS-Studio inputs integrate to EPICS via Py4Syn scripts to run scan routines and set up experiments.

Simplifying the experiment control system to a three-layered architecture, the EPICS channel access protocol works as an intermediate layer (or "*Service Layer"*). It transfers information between low-level device drivers and the high-level user interface. The graphical user interfaces and experiment command sequences stay on the top layer, or so-called *"Application Layer"*, assigning parameters in the form of EPICS PV's to EPICS IOC's (*Input-Output Controllers*). These IOC's on the "*Service Layer*" translate the parameters received from above into commands and signals to the device drivers on the layer below, the "*Driver Layer"*. To connect the PXI's FPGA environment to EPICS Records, an in-house data exchange protocol called Hyppie [2, 3] creates a bridge between EPICS and the low-level hardware control. The FPGA side runs on LabVIEW Real-Time, as the EPICS side runs on Red Hat Linux. Arriving commands from the application layer land at the Linux side, passing to the RT side over shared memory on the PXI chassis. The hard-

[1] gabriel.moreno@lnls

ISBN 978-3-95450-148-9

ware synchronization is set through 5V TTL signals, exchanged between the scan participants, and centralized at the NI PXI-6602 board. Figure 1 shows the IMX beamline context diagram for experiment control.

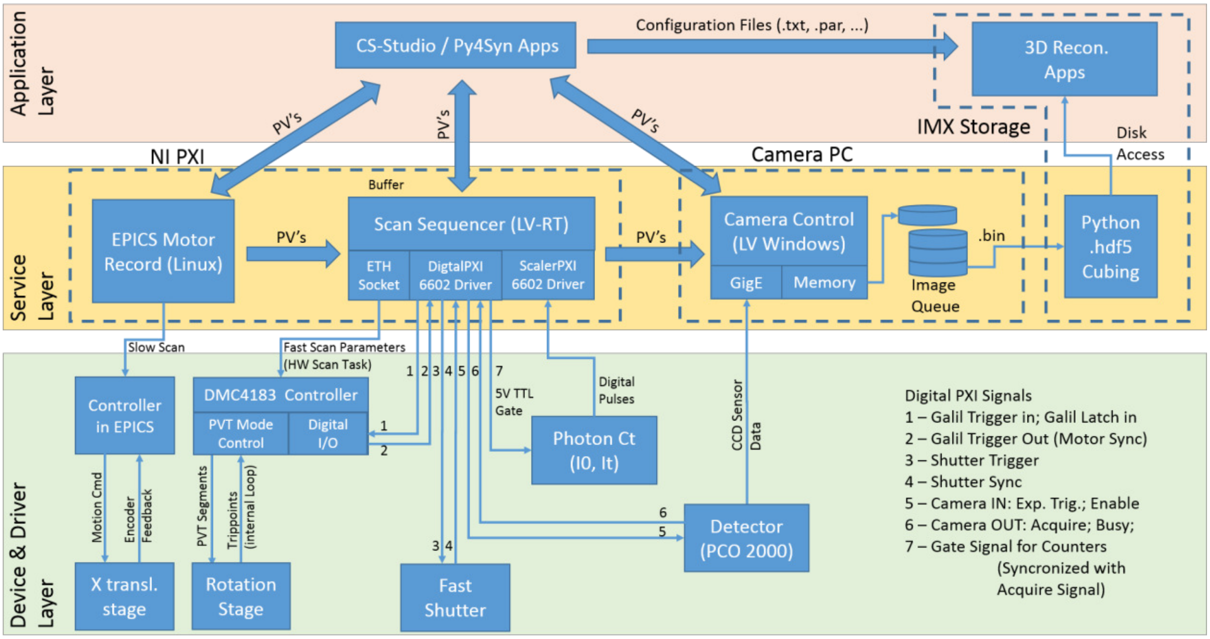

Figure 1: Data acquisition architecture using EPICS Records and LabVIEW drivers.

Scan Sequencer Under Hyppie System

To control the experiment via the NI PXI-6602 board, a new LabVIEW RT implementation developed as a Hyppie module controls the TTL signals triggering the instruments included in the scan task. Hyppie is a project conceived as collaboration between LNLS and NI Brazil to make a bridge between EPICS records and beamline devices connected to PXI-1045 chassis boards. There, the real-time Hypervisor runs both Windows-RT and Red Hat Linux, sharing I/O ports, RAM, and CPU cores. This way, EPICS IOCs, running on Linux, and LabVIEW VIs, on Windows-RT, can share information using a common memory.

The Scan Sequencer is a LabVIEW subVI implemented for handling the scan trigger sequence and send/receive feedback from the user in the form of EPICS PVs. It uses Hyppie's *ScalerPXI* and *DigitalPXI* modules to read and control the PXI-6602 board ports. An embedded EPICS Server in LabVIEW publishes the scan parameters in the form of changeable PVs. In the subVI, state machines run scan sequences oriented for each scan type in IMX, oriented by local variables containing captured PV values. After scan start, only acquired points publication and events expecting PV's for pausing scan are still active.

Synchronization between Hyppie System clock and the Scan Sequencer module proposes better performance scalability. In a performance upgrade opportunity, a reduction on Hyppie's clock period will affect all modules below, including the Scan Sequencer clock. The decision for centralizing trigger signals on the same board instead of connecting equipment in cascade also allows for modularity and scalability. On a device upgrade or exchange, for instance, as long as the new equipment is "triggerable", no reprogramming is necessary to integrate it into the system.

The sequencer is currently responsible for configuration of Galil DMC-4183 motion controllers via ethernet socket, using a VISA resource on LabVIEW. It receives parameters in the form of EPICS PVs and translates them into commands and parameters passed later by the socket.

Galil Controller Implementation

The DMC-4183 controllers, used for most of the IMX Beamline motors, allows the configuration of different motion modes, including Vector Mode, PVT Mode, and ECAM Table mode. The use of these motion modes cover both fly-scans and automated Point-to-Point scans. Sets of parameters passed coming from upper layer software configure the desired trippoints along the continuous or sequential motion path, changing the controller's operation according to each scan purpose. The use of *latch inputs* complements the trajectory programming by enabling position storing on acquisition start. At the IMX Beamline, the acquisition signal acts as a feedback for the DMC-4183 latch inputs, triggering the position storing event. Later, 3D reconstruction software can use the stored positions for error compensation. The execution splits into threads, simultaneously executing the latch position storage and the scan triggering. While the fast scan is not in use, the Galil controller switches back to the common mode of operation, disabling unnecessary threads, and re-enabling threads commonly used.

Camera Control Implementation

A dedicated machine running Windows OS controls the PCO 2000 camera through a GigE interface. The machine has two network boards to handle high data throughput during scans. A dedicated TOE (*TCP/IP Offload Engine*) network board receives data directly from the camera, and a second network card sends data to the data storage location. Once the data reaches the storage, other computers can remotely access it to process or analyze information. In the data route from the camera to the storage, the network configuration in test phase at IMX relies on big package transmission to minimize package loss and network latency. Package size and number of coalescence buffers are high, and all the network switches between the camera and the storage have QoS (*Quality of Service*) priority configuration. The storage is a GPFS file system with the purpose of providing better cost-effective scalability. With such configurations, the data transfer rate reaches 100 MB/s, or around 12fps at full resolution. On the Camera Control application, a queue system is included to account for any additional latency on the network.

In this implementation, a LabVIEW code configures the camera, translating PVs into camera inputs, and writes binary projection data to the storage location. Unnecessary interruptions and EPICS Client PV checks are disabled during tomography and re-enabled afterward. As a way to avoid unnecessary file conversion tasks, external scripts access the binary files from the storage location to convert data to HDF5, TIFF or other image format requested. For the purpose of tomography reconstruction, all images are stacked into a single HDF5 format image. In this way, it is possible to retrieve each projection for the various angles and extract the sinogram directly by changing the stack's cutting plane dimension.

For faster acquisition rates, a 1GB internal storage is available on the camera. However, given the flux and intensity currently available at the LNLS storage ring, the use of binning is necessary for acquisitions with an exposure time of less than 50ms. As binning reduces file size, in some settings it is also possible to store an entire scan on the camera's internal memory, thus removing the readout time incurred during image transfer. As the same applies to scans of regions of interest, the use of such combinations permits higher rate fly-scan testing at IMX Beamline.

CS-Studio / Py4Syn Applications

CS-Studio (Control System Studio), an open source toolset developed by Kay Kasemir of ORNL [5] is the main graphical interface of the beamline. It comprises a set of Eclipse tools that can interact with EPICS PVs. Through CS-Studio, users can view all necessary beamline components, setup scan parameters, and monitor scan progress through a single interface, as well as view the acquired images. For more elaborate routines, such as automated microscope focusing or sample and detector alignment, Python scripts handle and process raw data.

The new Python-based library, Py4Syn, provides a high-level abstraction for device manipulation, scan routines, interactive data fitting and plots [4]. While the scans are complex in nature, there is minimal interaction with the user is minimal, requiring a few parameters at most to run a full experiment. CS-Studio interface calls all python scripts to maintain consistency for the end user

EXPERIMENT SEQUENCE

The flowchart in Fig. 2 shows the process flow in the usual tomography experiment performed at IMX. To perform fast scans, the execution separates into EPICS control and Hardware Control, which runs the scan independently from the motor record. In this occasion, the motor record is used only for updating motor positions. Slow and non-sequential tasks stay controlled via EPICS while the sample rotation runs directly from DMC code. For frame rates higher than 5 Hz, the Uniblitz shutter control can be disabled to stay open during the entire acquisition.

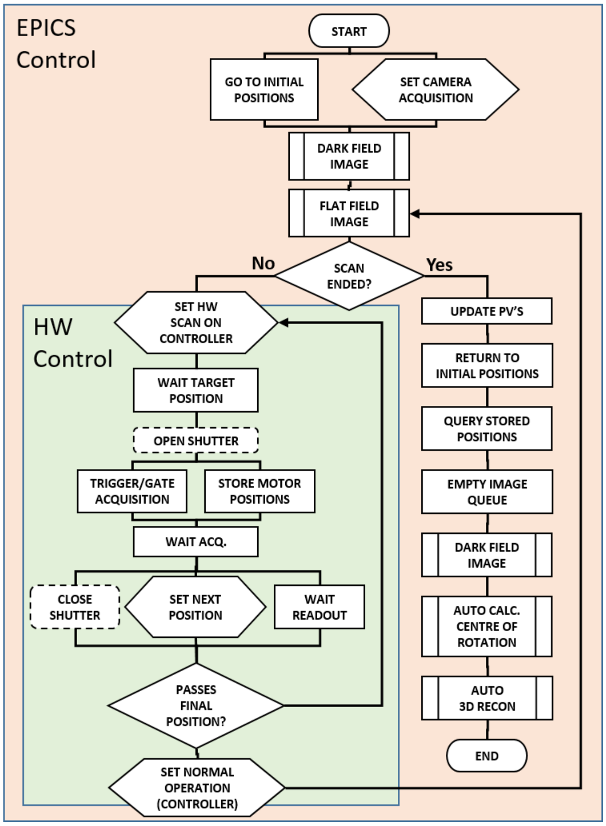

Figure 2: Fast Tomography Experiment Flowchart.

After the execution of the fast parcel, all PVs related to the motor record are updated. As soon as the scan ends, fully automated 3D reconstruction starts, using in-house software [6] or ESRF's PyHST2 [7] software. In-house development automatically calculates sample's center of rotation and the PyHST2 recon parameters are set following current system PV conditions.

EXPERIMENTAL RESULTS

The tests performed at IMX beamline comprised a 1000 projections tomography from 0 to 180 degrees, using 1x8 binning, 1 ADC, and high rate pixel clock on the PCO camera. This experiment configuration brought the acquisition and readout times to 10 ms, and 14.2 ms respectively. This way, a frame rate of 20 Hz or about ten times faster than normal acquisition in the same conditions was tested. Table 1 compares the performance of the three scan modes available at IMX Beamline.

Table 1: Trigger Times for IMX Scans

Times (ms)	Normal Scan	HW. Tr. Pt-to-Pt	HW. Tr. Fly-Scan
Exposure time	10 ms	10 ms	10 ms
Read-Out	14.2 ms	14.2 ms	14.2 ms
Motion	193 ms	71.0 ms	46.8 ms
Frame Period	409 ms	88.0 ms	49.2 ms
Latency	206 ms	4-6 ms	2.4 ms
Dead Time	399 ms	78 ms	39.2 ms

The Sample used for this fast tomography test was a common bamboo toothpick, with the reconstructed slices oriented in a cross-section along the direction of the bamboo wood fibers. The effective pixel size was 0,82 µm along the slice, and 6,56 µm vertically along transversal planes. Figure 3 shows the comparison between the slices obtained from the scans, with histogram comparison below.

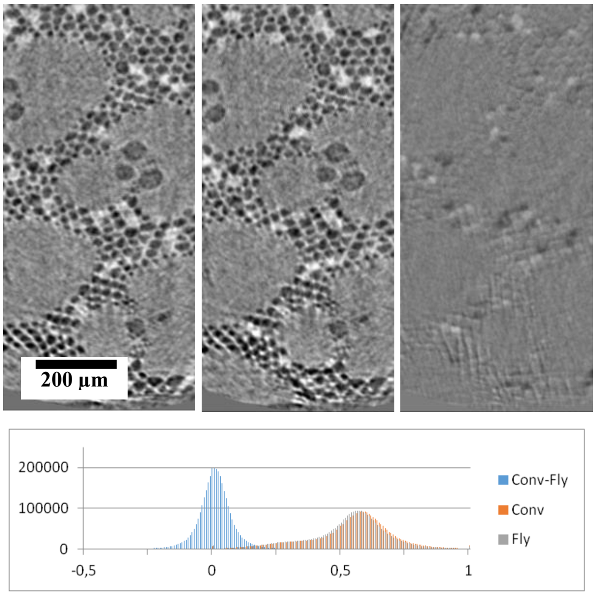

Figure 3: From left to right, normalized slices reconstructed from 8.5 min conventional tomography, 50 seconds Fly-Scan, and error between them.

As shown by the image on the right, the slices from continuous scans have comparable quality with the slices obtained from conventional point-to-point scans. Tests showed a reduction of 99% in the scan latency per point, and 88% decrease in the acquisition time, bringing capability closer to 4D experiments.

CONCLUSION

The tests proved the capability of fast and continuous scan alternatives, with comparable error and frame times at least one order of magnitude shorter than conventional scan methods. Short-term gain for IMX beamline includes fast low-resolution exploratory tomography, reduction of shifts per user, and applications closer to 4D tomography. For long-term applications in Sirius, one cannot disregard data throughput and motion system stabilization. Faster experiment execution will push the beamline infrastructure to its limits, and even sample loading efficiency will have to improve.

Future test phases in IMX involve integrating a new UPR160-Air rotation stage from PI-Micos, which should improve the speed capacity and motion stability for Fly-Scans. Planning includes regular improvements in the network capacity, a robotic sample changer, and even detector upgrades.

The system is modular and scalable, being already implemented and adapted to other beamlines at LNLS [8]. These qualities ensure the extension of this system to other LNLS beamlines shortly.

ACKNOWLEDGMENT

This work acknowledges the entire IMX beamline staff, SIL, GAE and SOL groups from LNLS for sharing ideas and making this work possible.

REFERENCES

[1] O'Dowd, F. et al., "X-ray micro-tomography at the IMX beamline (LNLS)", MEDSI 2014 Proceedings, Australia, 2014.

[2] Piton, J. R. et al., "Hyppie: A hypervisored PXI for physics instrumentation under EPICS", BIW 2012, Newport News, MOPG031.

[3] Piton J. R. et al., "A Status Update on Hyppie: A Hypervisored PXI for Physics Instrumentation under EPICS", ICALEPCS 2013, San Francisco, CA, USA TUPPC036.

[4] Slepicka, H. et al., 2015. "Py4Syn: Python for synchrotrons". *Journal of Synchrotron Radiation,* Volume 22, pp. 1182-1189.

[5] *CS-Studio Guide.* http://csstudio.sourceforge.net/docbook/css_book.pdf

[6] Miqueles, E.X. et al., "High Precision Algorithm for the Center of Rotation in Micro-Tomography", Tech Report, LNLS, 2015.

[7] Mirone, A. et al., 2014. "PyHST2 hybrid distributed code for high-speed tomographic reconstruction with iterative reconstruction and a priori knowledge capabilities". *Nuclear Instruments and Methods in Physics Research Section B,* Volume 324, pp. 41-48.

[8] Perez, C. A., et al., "Development of Fast Scanning X-Ray Fluorescence Microscopy at the LNLS D09B-XRF Beamline", ICXOM23, USA, 2015.