ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOBR04

GENERIC DATA ACQUISITION CONTROL SYSTEM STACK ON THE MTCA PLATFORM

J. Krasna, J. Varlec, U. Legat, Cosylab, Ljubljana, Slovenia

Abstract

Cosylab is the world leading integrator of control systems for big physics facilities. We frequently integrate high speed data acquisition devices on the MicroTCA platform for our customers. To simplify this process, we have developed a generic control system stack that allows us to support a large set of MicroTCA hardware boards with minimal firmware and software modifications. Our firmware supports generic data acquisition up to 32-bit sample width and also generic data generation. The firmware modules are implemented in a way so that support for MRF timing modules can be added and allow the board to act as a MRF timing receiver. On the software side we implemented the control software stack in NDS which means that we offer support for EPICS and TANGO control system out of the box.

SYSTEM DESIGN

Cosylab had worked on multiple high-performance data acquisition (DAQ) solutions throughout the years and some of those DAQ solutions, like applications for beam Current monitors, beam profile monitors, LLRF, etc. had overlapping core functionality. Because of this an challenge arose to determine if a generic control system stack that would cover this fundamental, overlapping functionality could be implemented.

The main objective was to design a system that would improve code reusability and reduce time to support new hardware without sacrificing the ability to extend functionality for custom use-cases. Additionally, the requirement was to cover as many control system frameworks as possible and make the system agnostic to specific frameworks. One of the desired outcomes was to reduce expertise needed to support existing and custom use-cases and create a central, standardized, mature codebase that all engineers could learn from.

Hardware

Looking at the generic form-factors currently available on the market that support DAQ use cases, especially for high-end applications, we can safely say that MicroTCA [1] has the most vendors supporting it and it is growing in popularity on the user-side as well. It is in use at different labs around the world for the already mentioned use cases (BCMs, BPMs, LLRF) and is commonly selected as the go-to platform for DAQ applications. The reasoning is high performance, customizability options through rear transition module expansions as well as advanced features, like IPMI [2] support and others. By working with multiple MicroTCA vendors, for example Cosylab and Teledyne developed DAQ drivers that are used at ITER [3], we have gained a lot of experience and insight into the platform. When selecting a hardware platform for our own product, a Dose delivery system used in medical particle accelerators for cancer treatment, the decision to go towards MicroTCA was a straightforward one.

Figure 1: Vadatech AMC523 module.

After careful consideration which hardware was most appropriate for our general DAQ system or gDAQ for short, we decided to use AMC523 [4] from Vadatech as well as the MRT523 rear transition module [5] and MZ523B mezzanine [6], see Fig. 1. The boards support 12 analog input channels with variable gain, 2 analog output channels and can sample at 125 megasamples per second.

Software

The application driver was implemented using NDS3 or Nominal Device Support [7] which is a public, open-source library that was developed at Cosylab to simplify integration of DAQ hardware for a variety of control system frameworks but is targeted mainly for EPICS [8] and TANGO [9]. The first iteration was built with the focus on EPICS, see Fig. 2. All GUI components were developed using Control System Studio Phoebus [10], see Fig. 3.

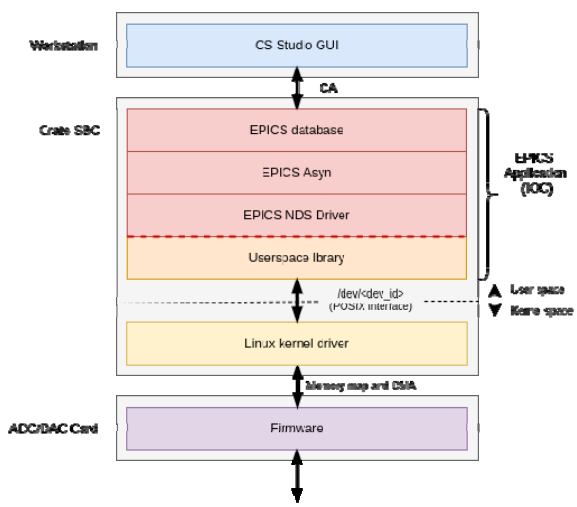

Figure 2: System architecture of the EPICS application.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOBR04

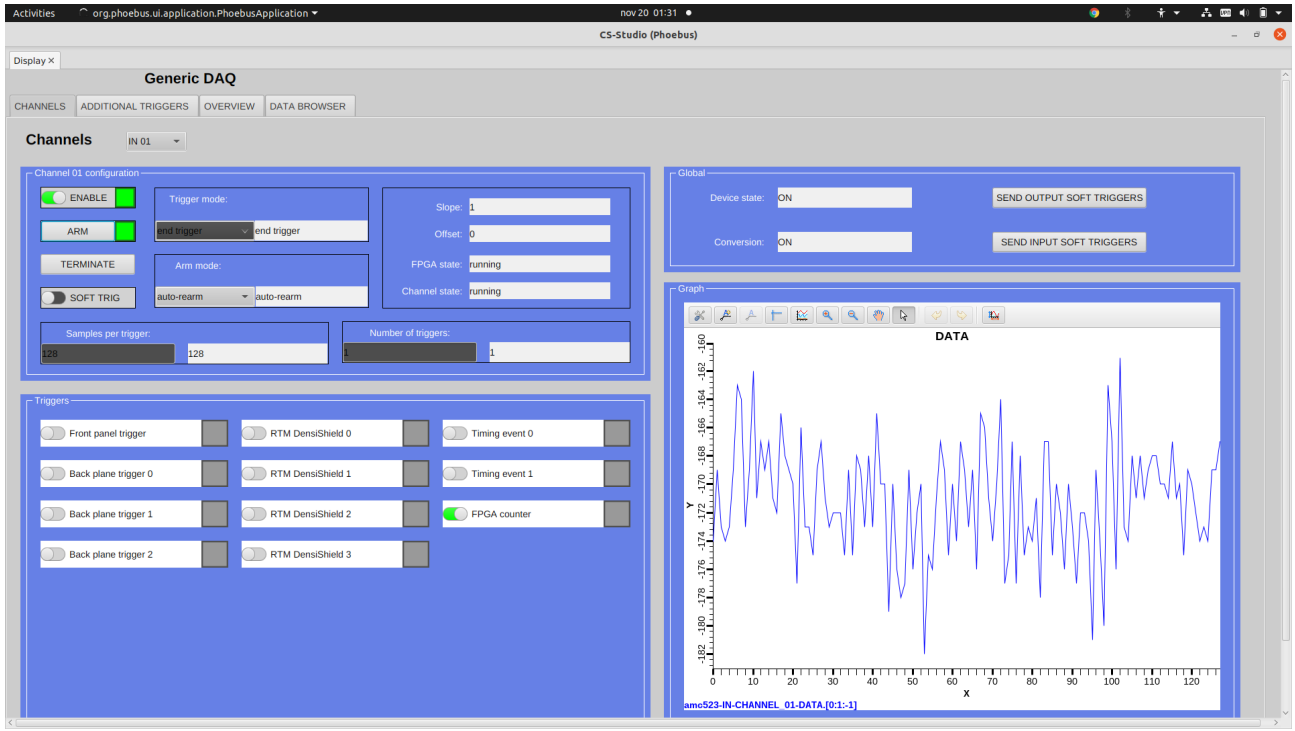

Figure 3: System GUI were implemented using the Control System Studio Phoebus.

In order to write a system that is hardware-agnostic and generalized we moved most hardware-specific parameters to a JSON [11] configuration file which is used by all components in the stack to adapt to a specific hardware board. The firmware is also configured by pushing the configuration data that is also stored in the JSON file through the userspace library.

The kernel driver was designed to include as little functionality as possible to reduce the amount of code running in the kernel and thus decreasing the likelihood of causing a kernel crash due to bugs or errors. The userspace library has a generalized but still rich and user-friendly C++ API which allows direct integrations, avoiding NDS altogether. The userspace library was written with extensibility in mind by exposing the firmware register set which makes adding support for new firmware logic simple and straightforward.

Firmware

Nominal Device Support provides a general abstraction of the device and its signals and while it supports both EPICS and TANGO out of the box it can be extended to support other frameworks as long as the tree-like description of device and its channels and the state machine logic are versatile enough for the new interface. The main feature of NDS that made this generic stack possible is that NDS generates the EPICS database and TANGO device servers programmatically which enabled using the JSON configuration as input during generation.

Firmware was designed and implemented to support a wide variety of DAQ features that can be found on firmware modules from other vendors:

- Configurable data width up to 32-bit
- Configurable up to 32 data acquisition and 32 signal generation channels
- Configurable buffer allocation for each channel separately
- Separate enable/disable for each channel
- Multiple trigger sources including software
- Arm and automatic re-arm
- Trigger mode: Front trigger, end trigger (prebuffering)
- Multiple data generation sources: from RAM, internal data stream (NCO), digitally processed ADC data
- Data streaming options: on trigger, periodic

We are also confident that firmware can be ported to other hardware platforms with minimal changes as long as a similar FPGA chip is available as the one used on the AMC523 (Kintex-7) although no such ports are planned as of this moment.

Timing Support

In order to support timing functionality an FPGA core was developed that implements Micro Reseach Finland (MRF) [12] receiver functionality. This core can be included on DAQ boards that provide an SFP or other compatible connector and in this way allow the timing to be connected directly to the board. The more typical use-case of having a separate MRF receiver and triggering DAQ over the backplane or through a front-panel connector is also supported out of the box.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOBR04

CONCLUSION

The gDAQ system was developed, tested and works as expected. We were able to cover all the generic functionality and therefore have a starting baseline for future projects. The system is currently in a prototype phase and was not yet deployed to a production environment.

REFERENCES

[1] MicroTCA standard overview, https://www.picmg.org/openstandards/microtca/

[2] IPMI Specification, V2.0, Rev. 1.1: Document, https://www.intel.com/content/www/us/en/products/docs/servers/ipmi/ipmi-second-gen-interface-spec-v2-rev1-1.html

[3] ITER Project, https://www.iter.org/

[4] Vadatech AMC523 DAC module, https://www.vadatech.com/product.php?product=384

[5] Vadatech MRT523 Rear Transition Module, https://www.vadatech.com/product.php?product=487

[6] Vadatech MZ523B 12 channel ADC module, https://www.vadatech.com/product.php?product=485

[7] NDS – Nominal Device Support v3, https://github.com/Cosylab/nds3

[8] EPICS – Experimental Physics and Industrial Controls System, https://epics-controls.org

[9] TANGO open-source SCADA and DCS, https://www.tango-controls.org/

[10] Control System Studio (Phoebus), https://controlssoftware.sns.ornl.gov/css_phoebus/

[11] JSON – JavaScript Object Notation, https://www.json.org/json-en.html

[12] Micro Research Finland Oy, http://mrf.fi