ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA215

A CONTROL ARCHITECTURE PROPOSAL FOR SIRIUS BEAMLINES

M. A. L. Moraes†, H. D. Almeida, R. M. Caliari, R. R. Geraldes, G. B. Z. L. Moreno, J. R. Piton, L. Sanfelici, LNLS, Campinas, Brazil

Abstract

With the increased performance provided by 4th generation synchrotron light sources, precise motion control and event synchronization are essential factors to ensure experiment resolution and performance. Many advanced beamline systems, such as a new high-dynamic double crystal monochromator (HD-DCM) [1], are under development for Sirius, the new machine under construction in Brazil. Among the expected performance challenges in such applications, complex coordinated movements during flyscans/continuous scans, hardware synchronization for pump-and-probe experiments and active noise suppression are goals to be met. Two architectures are proposed to cover general-purpose and advanced applications. The HD-DCM controller was implemented in a MATLAB/Simulink environment, which is optimized for RCP [2, 3]. Hence, its software must be adapted to a more cost-effective platform. One candidate controller is the NI cRIO. The portability of both MATLAB and NI PXI, the present standard control platform at LNLS, codes to cRIO is evaluated in this paper. Control resolution, acquisition rates and other factors that might limit the performance of these advanced applications are also discussed.

INTRODUCTION

Sirius is a 4th generation synchrotron light source [4] under construction in the Brazilian Synchrotron Light Laboratory, LNLS. With 3 GeV and low emittance (0.25 nm.rad) [5], it will provide one of the brightest synchrotron light sources in the world.

The LNLS Beamline Software Group is in a strategic moment to review the control system definitions, reevaluate which architecture best fits the new machine needs and define a new standard platform.

In LNLS, two categories of control systems are devised, namely:

- General-purpose control;
- Advanced applications control.

This paper presents an ongoing development whose main goal is to create a robust standard for control systems, minimizing heterogeneousness and enabling fine tuning on controller parameters. For each of these categories, one case is respectively presented:

- cRIO-Linux Project – migration from National Instruments (NI) PXI chassis to NI CompactRIO (cRIO);
- HD-DCM Tool – migration from Speedgoat xPC chassis to cRIO.

cRIO performance tests have been satisfactory and makes it a candidate for both general-purpose and advanced control new standards, offering a large portfolio of I/O modules and some robust software solutions under constant improvement by a large community.

GENERAL-PURPOSE CONTROL

History

In the 90's LNLS created its own hardware for data acquisition and motion control, due to industrial policies and budget constraints [6]. LOCO represented a big technological advance at the time and lasted as the main solution for control systems until 2013.

In a process of upgrade of the beamlines, a new control system standard was defined, which was intended to minimize efforts of maintenance by using solid and widely diffused solutions. This represented a big strategic change. EPICS [7], a world-wide standard, was chosen as middleware, due to its open-source continuous improvements and large collaborative community. The remaining decision to be made was the hardware platform. Although, there was no hardware platform integrated with Linux OS in National Instruments portfolio, NI engineers proved that PXI was a powerful platform and it was decided that starting a collaborative effort to make it compatible with EPICS was worthwhile. Hence, a project named HYPPIE [6] was developed by LNLS Beamline Software Group and NI to allow:

- dual operational system (OS) execution inside NI PXI via NI RT Hypervisor [8] – LabVIEW RT, for accessing PXI hardware, and Linux, for hosting EPICS server;
- DMA (direct memory access) between OSs, to link EPICS to PXI I/O.

Presently, commercial motion controllers (mostly from Galil and Parker) and data acquisition hardware together with HYPPIE compose the current LNLS standard general-purpose control system.

A New Standard Platform

Sirius brightness will be up to 1 billion times more intense than the operating light source (UVX) [9]. This huge increase demands, among other higher performance requirements, more stable instrumentation, meaning that, in terms of control system requirements, faster feedback sampling and more flexible controller configurations shall be common.

At this point, the first HYPPIE limitation arises. The dual OS feature is based on NI RT Hypervisor, whose development was discontinued. Therefore, the effort to support new OSs ended [10], and neither NI LabVIEW RT nor Linux could be upgraded anymore. Another concern is about costs. FPGA would be an appropriate solution for latency reduction, but the FPGA module for PXI

† marcelo.moraes@lnls.br

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA215

is expensive enough to be considered feasible only for specific applications, not as a standard platform. Without FPGA, latency is limited by the RT OS. Finally, since the general-purpose Linux OS (non-RT) can't be changed, there are limitations about reusing IOCs (input/output controllers), due to incompatibilities as, for instance, the version of the compiler or the system architecture (x86).

Facing these points, HYPPIE is insufficient for advanced applications. Thus, keeping it for general-purpose systems would lead to a more heterogeneous standard architecture. Although, NI PXI is a powerful platform that can achieve Sirius' requirements with FPGA modules, it would demand high investments and, even the earlier versions, could not run Linux natively [11]. Hence, the new standard control system needs another solution.

The motivation for the HYPPIE project was the integration of robust hardware and Linux environment. In parallel with this implementation with the PXI at LNLS, NI announced in 2013 a completely redesigned cRIO with built-in Linux RT OS [11]. This is a very convenient option because cRIO is a rugged platform designed for industry and widely used around the world. With a built-in Linux RT, it is a robust base for integrating hardware and EPICS. Table 1 presents a comparison between HYPPIE and cRIO, with the purpose of emphasizing the features of interest for the new standard definition. Despite the unfairness of comparing recent technology like the TSN (Time Sensitive Network) [12] with a 5-year-old system, deterministic network is an interesting feature for advanced applications.

Table 1: Comparison between HYPPIE and cRIO

Feature	PXI AND HYPPIE	cRIO[1]
RT OS	Phar Lap ETC [11] that cannot be upgraded	NI Linux RT [11]
EPICS	Additional Virtual Machine with non-RT OS	Compatible with native RT OS
TSN [12]	-	Compatible
FPGA	Expensive modules	In-built
Scalability	Limited by virtualization technology	Scalable
Hot swappable [13]	No	Yes

Unlike the relevant effort of creating a standard control system by migrating HYPPIE tasks to cRIO, restricting the options for motion controllers might be unwise or unfeasible. General-purpose motion controllers can offer a satisfactory solution for a large volume of stepper and servo axes at significantly lower cost. Indeed, simple instruments, based on average-performance drivers and actuators, controlled via PID, notch and low-pass filters [14], make most of the motion applications. Galil 4183 controller not only offers such functionalities [15] at a reasonable cost, but it has also been satisfactory used for some years. Therefore, it will be kept as standard motion controller. In addition to it, a few controllers that may be linked with specific third-party motors and actuators will also be inevitably present in a smaller amount. Finally, customized controller implementations shall be categorized as advanced control solutions, due to requirements of plant identification, bandwidth and fast I/O.

cRIO-Linux Project

cRIO-Linux is a multi-layer solution for integrating cRIO I/O to EPICS, minimizing latency and maximizing throughput. The first phase of the project consists in defining the complete architecture. The second phase is the implementation of read/write hardware features that go from basic to advanced implementations.

The proposed features to cover all general-purpose control systems that use the cRIO platform are:

1. Digital read/write (arrays, FIFO);
2. Analog read/write (arrays, FIFO);
3. Scaler;
4. Encoder in;
5. Stepper motor out;
6. RS232/RS485 2-wires/RS485 4-wires Serial Port;
7. EtherCAT devices;
8. Triggering.

Architecture Figure 1 presents the architecture of a general-purpose control system on cRIO platform. Considering that this project consists in hardware migration, it is possible to reuse the existing GUI structure. The high-level layers are based in Python, whose implementation is detailed in [16]. Although the architecture is represented from hardware to GUI, the object of study of this paper ends in the EPICS layer.

cRIO-Linux Project is composed by four layers: FPGA, Real-time, Cpp libraries and Device support (green blocks). In cRIO architecture, every hardware is accessed through FPGA, and there are tools for access to FPGA and Real-Time applications, as well. LabVIEW FPGA C API [17] is a solution for direct access to FPGA via C code, whereas LabVIEW Real-Time applications can interact with external applications via POSIX IPC [18]. This way, all requirements for cRIO-Linux Libraries (Libs) layer development are satisfied. The chosen language is Cpp, for object-oriented implementation.

Development Status The first phase is finished and the second phase is in progress, having the digital read array already successfully implemented in all layers. The development has been done by feature, which means that, according to the architecture in Figure 1, the architecture has been vertically iterated for each feature until conclusion.

[1] Model NI-cRIO 9039

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA215

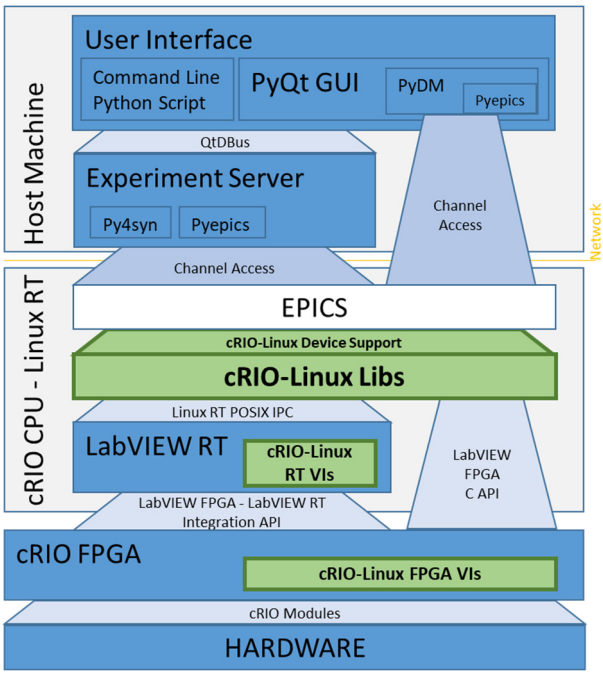

Figure 1: General-purpose control systems standard architecture based on cRIO platform and LabVIEW libraries. cRIO-Linux Project layers are shown in green.

ADVANCED APPLICATIONS CONTROL

The Sirius project leveraged the creation of a new category of control system for beamline applications in LNLS. In the new high-end systems, there is no room for random effects, i.e. a deterministic design approach is essential and every aspect must be considered. Simply trusting hardware specs and/or theoretical performances is, unfortunately, not sufficient to achieve the required optimized performances. Hence, it is necessary to perform system identification routines to validate all the real performances of the elements in the control loop systems.

This is the point in which common motion controllers fail, even if their feedback sampling rates were ideal. Although a linearized system tends to be controlled by simpler control techniques, such as PID, having them as the only choice due to controller limitation is not desirable. The characteristics that differ general-purpose control systems from advanced applications control at LNLS can be summarized as follows:

1. Well-known system plant obtained from system identification procedure [14];
2. Fully runtime customizable controller, parameterized by transfer-function polynomials or state-space matrices;
3. Control loop feedback sampling above 10 kHz;
4. Need of robust enough software to use the full capacity of the hardware (hardware as bottleneck).

Another feature of interest for the advanced applications standard solution is hard real-time through EPICS, to enable inter-instruments deterministic interaction. According to [19], it is possible to achieve hard real-time execution with EPICS. Better results for latency shall be achieved via deterministic networks like TSN – built-in for some cRIO models. Hence, the cRIO flexibility makes it eligible as the standard platform for hosting also the advanced applications.

HD-DCM

Figure 2 shows the HD-DCM and its dynamic concept, aiming at a higher stability performance [14]. The schematic shows the first set of crystals attached to the Metrology Frame (MeF1), and the second set of crystals in the Short-Stroke (ShS). The motion system that controls the Bragg angle, parallelism and gap is composed by the actuators and sensors presented in

Table 2. The Goniometer Frame (GoF) motion defines the Bragg angle, the Long-Stroke (LoS) performs coarse gap movement and the ShS guarantees fine gap and parallelism.

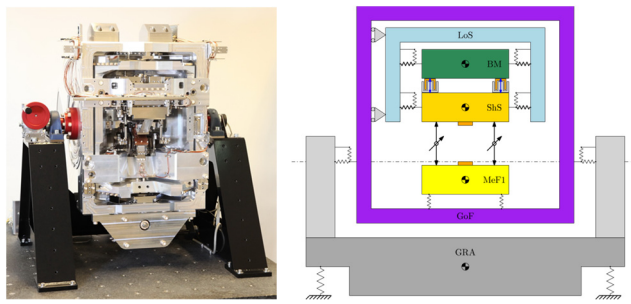

Figure 2: Left: The high-dynamic double-crystal monochromator (HD-DCM); Right: HD-DCM schematic.

Table 2: HD-DCM Core Motion Elements [21] and Feedback Requirements

Frame	DOF	Actuator	Sensor	FB sample rate
Gonio (GoF)	Rx	Torque motor	Rot. Incr. Encoder	10 kHz
Long-Stroke (LoS)	Y	Stepper	Lin. Abs. Encoder	10 kHz
Short-Stroke (ShS)	Y, Rx, Rz	3 x Voice-Coil	3 x IFM	20 kHz

The HD-DCM tool is a customized controller for the HD-DCM. It is an example of advanced control application which was successfully implemented on Speedgoat xPC [20]. Once the concepts were validated, the system started to be migrated to cRIO.

HD-DCM Tool in Speedgoat xPC

Speedgoat was the chosen platform for rapid control prototyping (RCP), due to its integration with MATLAB/Simulink RT. However, it is not a platform suited to the beamline environment, due to the availability of more cost-effective solutions.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA215

Despite the bottlenecks faced for BiSS latency and PWM update, the Speedgoat xPC results for system identification tasks and control feedback loop rates in real-time environment were remarkable [2].

Architecture The motion system architecture is presented in Figure 3. Using a custom implementation for motion control provided flexibility for the controller design. Moreover, using the same hardware in the whole development process, including delay and noise evaluation, and plant identification, may lead to more accurate results. Hence, stability was achieved within the design targets, validating mechatronic project concepts [14].

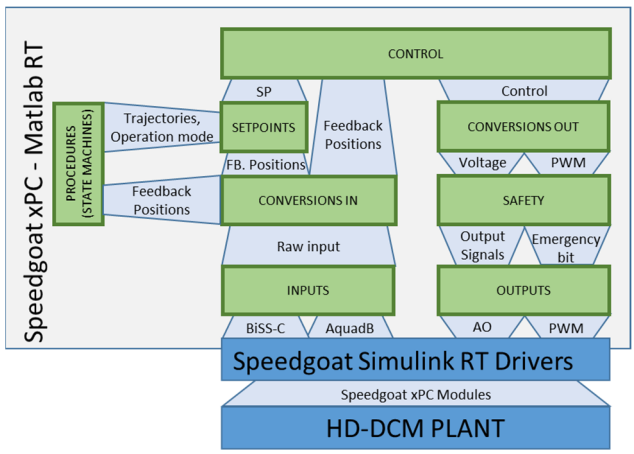

Figure 3: HD-DCM Tool (green) motion controller architecture based on Speedgoat xPC platform.

HD-DCM Tool Migration to cRIO

After the control prototype validation, the migration to a more appropriate platform has started. Among others, the reasons that make cRIO a more suitable hardware platform to the beamline environment are:

1. Price – cRIO offers a more cost-effective prospect;
2. Support – NI has offices in Brazil and solutions are typically quickly released;
3. Robustness – cRIO is made for industry and it is used even at extreme environments, such as oil platforms;
4. Large community – cRIO is used by a wide community, leading to more stable solutions;
5. Signal conditioning – cRIO offers much better solutions.

National Instruments offers some tools for running external models inside LabVIEW. Therefore, as a first approach two tools were tested in an attempt of reusing the original Simulink code:

- With NI VeriStand [22] it should be possible to run Simulink model compiled as a DLL file. However, NI software incompatibilities with Microsoft Windows 10 OS were found. The tutorial for VeriStand Model Compilation included installing Windows SDK 7.1 instead of the Windows SDK 10, which, on the other hand, is required for Simulink RT compilation [23, 24]. In addition to this annoying downgrade, which could interfere in Simulink RT compilation, a more severe limitation was noticed, namely: a benchmark of LabVIEW RT at cRIO was made under NI support and the conclusion was that the most powerful cRIO cannot reach 20 kHz in hard real-time application. The CPU usage was huge, even for a simple application. The recommended hard real-time rate of only 10 kHz is insufficient for the ShS closed-loop controller. In summary, cRIO could not achieve the impressive Speedgoat real-time rate, which is required by the HD-DCM.
- The NI Control Design and Simulation Module [25] has a model converter tool for converting external models into LabVIEW blocks. After some attempts, several unsupported or partially supported blocks were found in HD-DCM Simulink code [26, 27].

Once the Simulink code could neither run as a DLL in LabVIEW nor be converted to LabVIEW code, the remaining option was to reimplement it on cRIO platform.

Architecture Figure 44 depicts the architecture for the HD-DCM on cRIO. As the ShS controller requires a 20-kHz control loop rate, but LabVIEW RT is limited to 10 kHz, the ShS control loop, including kinematics transformations, must be implemented in the FPGA layer.

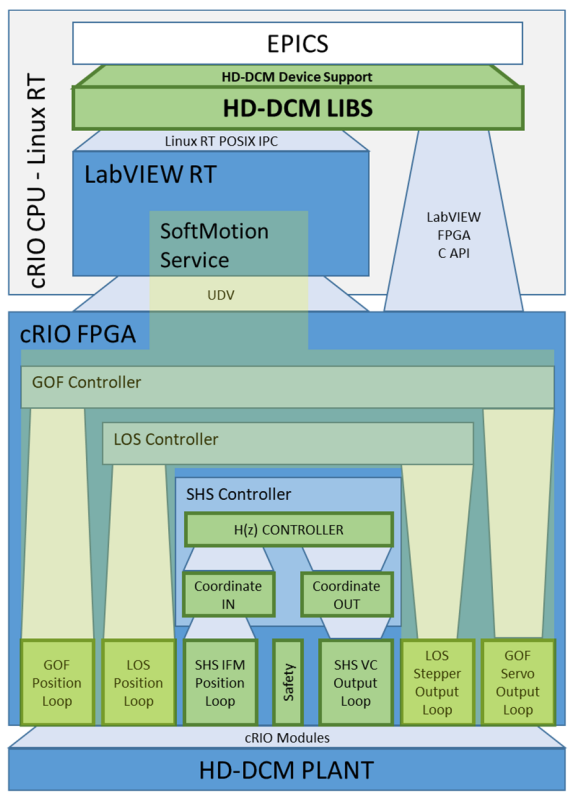

Figure 4: HD-DCM tool motion controller architecture based on cRIO platform.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA215

Conveniently enough, cRIO offers a set of motion control solutions (NI SoftMotion [28], dedicated 3rd party I/O modules) that covers trajectory generation, interpolation, velocity calculation, among other features for controlling stepper and servo motors. SoftMotion Service manages both FPGA and RT layers and will be used as the standard solution for customizable controllers.

Development Status SoftMotion configuration is in progress, whereas the I/O drivers and the ShS control loop with kinematics transformations are already finished. Thanks to the FPGA implementation, the ShS controller could reach 1 MHz. The analog output module (NI 9264), however, limits the rate to 25 kHz. Therefore, the hardware limitation target was achieved.

NEXT STEPS

The general-purpose control systems standard implementation is in the second phase. Once it is finished, the next step is the definition of a hardware solution for synchronization, triggering and counting for beamline timing management. In next months, a proof of concept is expected, with commercial tools to guide this decision. A test shall evaluate TSN low-latency deterministic and synchronous network characteristics. If satisfactory latency (for sampling time around 10 kHz) is reached, an integration between beamline instruments beamline shall be developed. As an example, the undulator and a beam position monitor are expected to work as setpoint and feedback position signals, respectively, for the HD-DCM. Also driven by the DCM, temperature control will be soon addressed, including cryocooling, heaters and temperature sensors. Finally, the different control loops, presented in

Table 2, will be implemented in cRIO and must have their performance compared with the Speedgoat xPC results.

CONCLUSION

Defining a new standard is an important task that affects the routines of support groups for years. Wrong decisions demand rework that results in higher costs and heterogeneous solutions.

The present study proposes architectures for control systems of distinct levels of complexity and a standard hardware platform to host these applications, which even prepares the standardization of network and beamline time-synchronization tools. NI cRIO is a candidate platform to standard control system, satisfactory as for general-purpose controllers as for advanced applications, covering all aspects for a lower latency EPICS integrated solution. It shall enable the creation of a homogeneous control system.

To conclude, working with controller state-space models – as proposed in advanced applications architecture – proved to make migrations, when necessary, very flexible, because they are not tied to a specific commercial controller.

ACKNOWLEDGEMENTS

The authors would like to gratefully acknowledge the funding by the Brazilian Ministry of Science, Technology, Innovation and Communication and the contribution of the LNLS team, the National Instruments team, the MI-Partners team, and those of the synchrotron community who directly or indirectly built the path to this development.

REFERENCES

[1] R. R. Geraldes, R. M Caliari, M. Saveri Silva, "Intrumento para movimentação e posicionamento de elementos ópticos com resolução e estabilidade mecânica nanométricas em li-nhas de luz", PCT/BR2017/050262, 2017, Patent Applica-tion.

[2] G. B. Z. L. Moreno, R. M. Caliari, R. R. Geraldes and M. A. L. Moraes, "Rapid Control Prototyping Tool for The Sirius High-Dynamic DCM Control System", presented at ICALEPCS 2017, Barcelona, Spain, Oct 2017, paper THPHA214, this conference.

[3] Rapid Control Prototyping (RPC) - Speedgoat, https://www.speedgoat.com/applications-industries/applications/controller-prototyping

[4] A. R. D. Rodrigues *et al.*, "Sirius Status Report", *in Proc. IPAC'16*, Busan, Korea, May 2016, paper WEPOW001, pp. 2811-2814.

[5] L. Liu, F. H. de Sá, and X. R. Resende, "A new optics for Sirius," *in Proc. IPAC'2016*, pp. 3413–3415.

[6] J. R. Piton *et al.*, "HYPPIE: A Hypervisored PXI for Physics Instrumentation under EPICS."

[7] EPICS – Experimental Physics and Industrial Control System, http://www.aps.anl.gov/epics

[8] NI Real-Time Hypervisor Architecture and Performance Details, http://www.ni.com/white-paper/9629/en

[9] The Sirius Project, http://lnls.cnpem.br/en/sirius-en/sirius-project

[10] What Hardware and Software is Supported for Use with NI Real-Time Hypervisor Systems?, http://digital.ni.com/public.nsf/allkb/6BFC1E669D4F1DED862575AF0073B7F1

[11] Real-Time Controllers and Real-Time Operating System Compatibility, http://www.ni.com/product-documentation/53636/en

[12] Time-Sensitive Networking Task Group, http://www.ieee802.org/1/pages/tsn.html

[13] Assemble Your CompactRIO System, https://www.ni.com/getting-started/set-up-hardware/compactrio/assemble

[14] R. M. Caliari, *et al.*, "System Identification and Control for the Sirius High-Dynamic DCM", presented at ICALEPCS 2017, Barcelona, Spain, Oct 2017, paper TUSH203, this conference.

[15] DMC-41x3 - Manual Rev. 1.0k, http://www.galilmc.com/download/manual/man_41x3.pdf, pp. 175

[16] G. S. Fedel, D. B. Beniz, L. P. Carmo and J. R. Piton, "Python for User Interfaces at Sirius", presented at ICALEPCS 2017, Barcelona, Spain, Oct 2017, paper THAPL04, this conference.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA215

[17] Introduction to the FPGA Interface C API, http://www.ni.com/product-documentation/9036/en
[18] NI Linux RT Libraries, https://forums.ni.com/t5/Reference-Design-Content/NI-Linux-RT-Libraries/ta-p/3622838
[19] A. Barbalace et al., "Performance Comparison of EPICS IOC and MARTe in a Hard Real-Rime Control Application.", *IEEE Trans. Nuc. Sci.*, vol. 58, pp. 3162-3166, 2011.
[20] Speedgoat Performance - 19" Simulink real-time test computer (PC), https://www.speedgoat.com/products-services/real-time-target-machines/performance
[21] R. R. Geraldes, R. M. Caliari and M. S. Silva, "Método de controle de grau de liberdade em sistemas mecatrônicos e monocromador de duplo cristal." INPI BR 10 2016 020900 5, 2016.
[22] VeriStand - National Instruments, http://www.ni.com/veristand
[23] Setting up The Mathworks, Inc. MATLAB® Software to Create a NI VeriStand or Model Interface Toolkit Compatible DLL, http://digital.ni.com/public.nsf/allkb/4AD1B9CC08DA876B86257F9B00531D4E?OpenDocument#2014
[24] Problem Installing the Windows SDK 7.1 for VeriStand Model Compilation, http://digital.ni.com/public.nsf/allkb/DD49A2842454008A86257F86004804FE?OpenDocument
[25] Using the Simulation Model Converter (Control Design and Simulation Module), https://zone.ni.com/reference/en-XX/help/371894H-01/lvsimconcepts/sim_c_translator
[26] Partially Supported Blocks (Control Design and Simulation Module), https://zone.ni.com/reference/en-XX/help/371894H-01/lvsimconcepts/sim_partial_support
[27] Unsupported Blocks (Control Design and Simulation Module), https://zone.ni.com/reference/en-XX/help/371894H-01/lvsimconcepts/sim_c_ublocks
[28] NI LabVIEW NI SoftMotion Module, http://www.ni.com/download/labview-ni-softmotion-module-2017/6786/en