TESTBED − AUTOMATED HARDWARE-IN-THE-LOOP TEST FRAMEWORK

P. Maslov, K. A. Meyer, K. Žagar, Cosylab, Ljubljana, Slovenia

Abstract

In a big physics facility such as ITER or ESS, the control system is typically updated at least 3 times a year. This means that prior to each minor release all components should be tested. For testing DAQ drivers, a test plan should be written, based on which a manual test is performed. The idea behind the TestBed suite is to execute tests automatically. Our TestBed is a PXI chassis which contains an embedded controller running the CODAC control system on a Scientific Linux operating system and a DAQ board capable of generating and acquiring analog and digital signals. It provides an easy-to-use framework written in Python and allows for the quick development and execution of automatic test scripts.

ARCHITECTURE

From the hardware perspective, each system under test (SUT) is physically connected to the TestBed (TB) (Fig. 1) with a connector board using a predefined pin configuration. Both SUT and TB are connected to the LAN (not shown in Fig. 1).

The software part consists of three tiers:

1. Software that provides the desired functionality of a DAQ board:
   - C executables
   - EPICS device support (NDS [1] driver + IOC)
   - *LabVIEW interface*
2. Python bindings in the form of a class that reflect the given functionality of 1)
3. Automatic test cases written by the test-plan engineer using 2).

The NI-PXI6259 functionality that is supported in the TB suite includes:

- Analog input/output (static) on a desired channel
- Analog input (waveform) on a trigger
- Analog output (waveform – sine/saw/square/from file) on a trigger signal
- Configuration of the DIO port mask
- DIO diagnostics: port mask (0 – input, 1 – output) and lines state (0 – low, 1 – high)
- Digital input/output (static) on a desired line
- Device reset

The underlying connection protocol is SSH and is provided by the Python package Paramiko (the NDS implementation utilizes the Python package CaChannel).

TESTBED BASE CLASS

The Testbed Python class (Fig.2, left hand-side) provides a set of test methods to be executed on an SUT. These methods can be extended for any DAQ board and then used to quickly write automatic test scripts.

An example of such test case script showing three tests for the NI 6259 DAQ board (Fig.2, right hand-side) is:

1. test_aio_static: generate static signal on AO0 (TB), acquire static signal from AI0 (SUT), compare results.
2. test_dio_static: generate static signal on DO0 (TB), acquire static signal from DI0 (SUT), compare results.
3. test_aio_wf: generate a sine wave on AO0 (TB) on the trigger signal PFI1, acquire the waveform on AI0 (SUT) on the trigger signal PFI1, compare results.

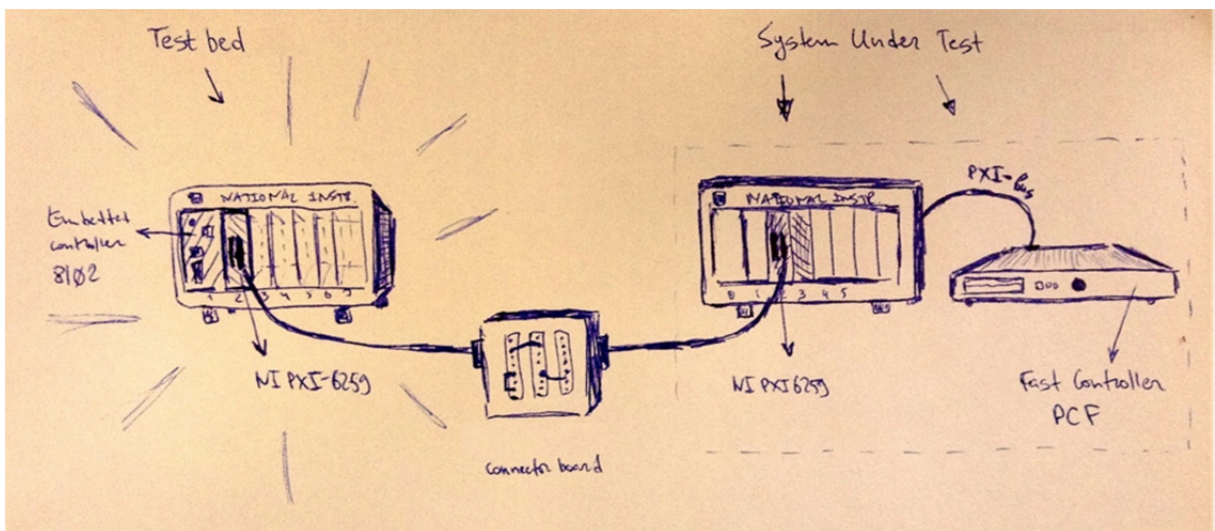

Figure 1: TestBed chassis is attached to the system under test.

ISBN 978-3-95450-146-5

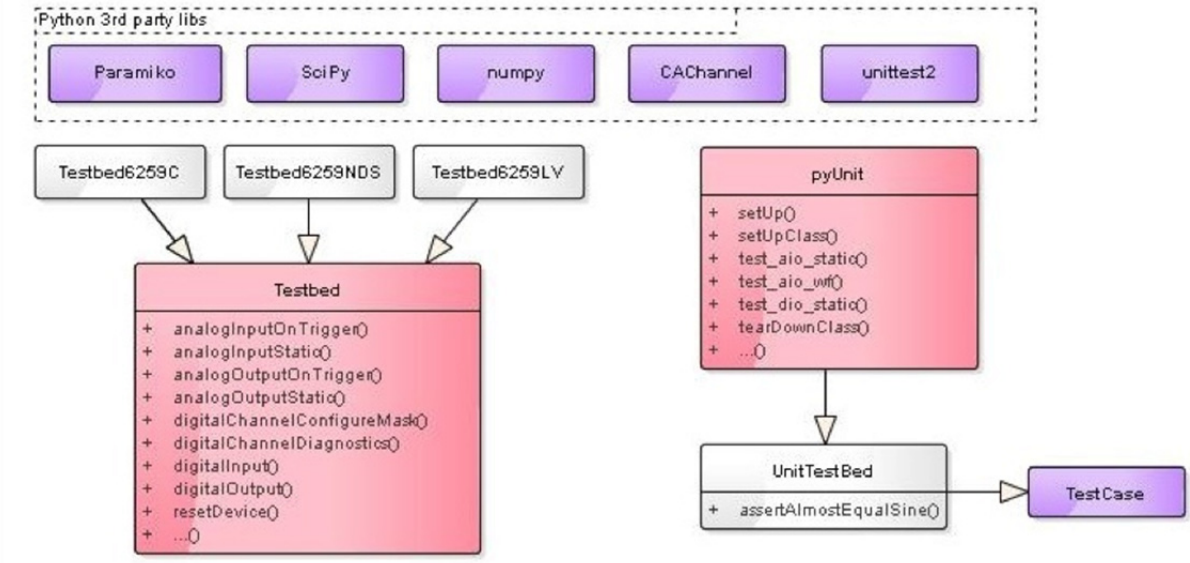

Figure 2: Python class diagram.

IMPLEMENTATIONS

C Executables

The functionality of a DAQ board can be supported by providing the C executables. In this work we have used the NI PXI-6259 Linux Device Driver [2], which provides the support library API to be used for application development.

NDS

If EPICS device support is available for the selected DAQ, then you no longer have to deal with files (saving and parsing waveforms from files, transferring files via SSH, etc.), but with EPICS Channel Access [3] (CA) – a protocol that provides remote access to records and fields managed by IOC.

The process can be simplified if the device support is written using NDS. NDS generalizes EPICS device support for data acquisition and timing devices. It provides sets of interfaces, solutions and best practices of device integration for EPICS. In this case, the Python code becomes identical for all DAQ cards. Which means that the only thing you need to do is provide an automatic test script (e.g., using the unittest2 Python package).

LabVIEW

Often, there is no Linux driver that supports the full functionality of a DAQ board. NI DAQ cards, however, come with the NI-DAQmx driver, that does support everything. In this case, the developer can implement something similar to NDS in LabVIEW, thus generalizing data acquisition cards by running an IOC and exposing NDS-like EPICS PVs to CA clients (including TestBed).

SUMMARY

- The PXI chassis, Embedded Controller and DAQ board have been selected for the TB suite.
- Scientific Linux 6.3 and ESS CODAC v4.1 were installed on the Embedded Controller.
- The CODAC pxi6259 example programs (written in C), EPICS device support (NDS) were modified to provide basic functionality – DIO, AIO, triggers, reading/writing data from/to files (PVs).
- The TestBed class was written in Python, and wraps the C and NDS functionality of the NI-PXI6259 DAQ board.
- The TestBed was tested on ITER CODAC v4.2 (Red Hat Enterprise Linux 6.3) and ITER CODAC v4.1 (Scientific Linux 6.3).
- The resulting test framework makes it possible for automatic tests to be executed with each release of the CODAC control system, thus reducing effort and ensuring complete consistency and repeatability in the testing protocol.

ACKNOWLEDGEMENT

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no 289485.

REFERENCES

[1] V. Isaev, "Nominal Data Acquisition Device Support for EPICS", Proc. ICALEPCS2013, TUPPC059, http://jacow.org/.

[2] K. Meyer *et al.*, "Design and implementation of Linux drivers for National Instruments IEEE 1588 Timing and General I/O cards", Proc. ICALEPCS2013, THPPC056, http://jacow.org/.

[3] http://www.aps.anl.gov/epics/docs/ca.php

ISBN 978-3-95450-146-5