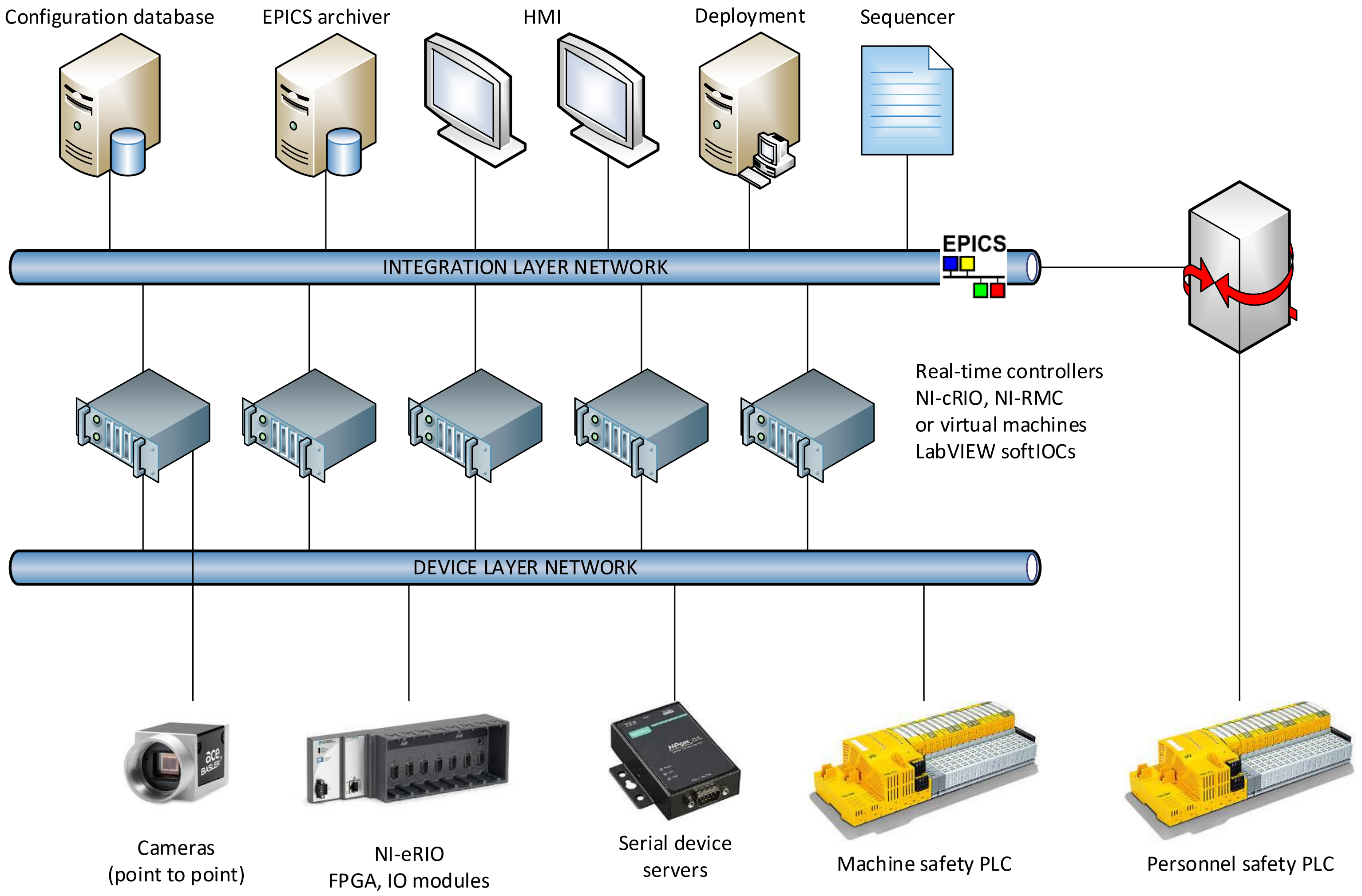
Integration of EPICS to NI platform

Karel Majer, Jan Fara

Motivation


Configuration database
EPICS archiver
HMI
Deployment
Sequencer
EPICS
INTEGRATION LAYER NETWORK
Real-time controllers
NI-cRIO, NI-RMC
or virtual machines
LabVIEW softIOCs
DEVICE LAYER NETWORK
Cameras
(point to point)
NI-eRIO
FPGA, IO modules
Serial device
servers
Machine safety PLC
Personnel safety PLC

Motivation

- (2017) Control system built in LabVIEW
  - Real-time code running on Pharlap OS
  - RT <-> GUI communication provided by NI Network Streams
  - Channel Access used for data archiving

- Existing libraries
  - NI EPICS VIs – missing record fields, problems with archiver
  - CALab – doesn’t run on Pharlap, NI RT Linux support added only recently

LabIOC

- Implementation of Channel Access server and client purely in LabVIEW
- Observatory Sciences + ELI
- Compliance with Channel Access protocol, compatible with EPICS 3.14 and EPICS 7

+	−
Does not use EPICS Base	Does not use EPICS Base
Runs on any LabVIEW target	Does not have full EPICS functionality

LabIOC server

Implemented

- Access to EPICS database from LabVIEW
- Basic data types (ai, ao, bi, bo, longin, longout, mbbi, mbbo, stringin, stringout, waveform)
- Alarms
- Monitor posts

Missing

- Database links
- Soft record types (calc, fanout etc.)
- Callbacks (run VI when records processes)
- Access control

LabIOC client

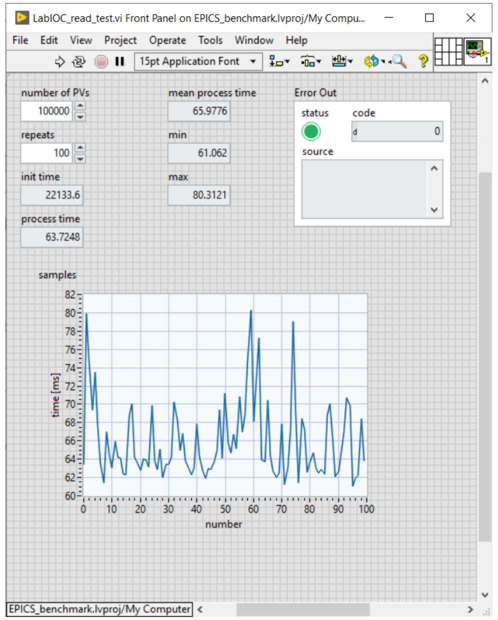
Implemented

- caget, caput, camonitor
- Batch operations on many PVs
- Automatic channel reconnect
- All Channel Access data types

Missing

LabIOC client benchmarks

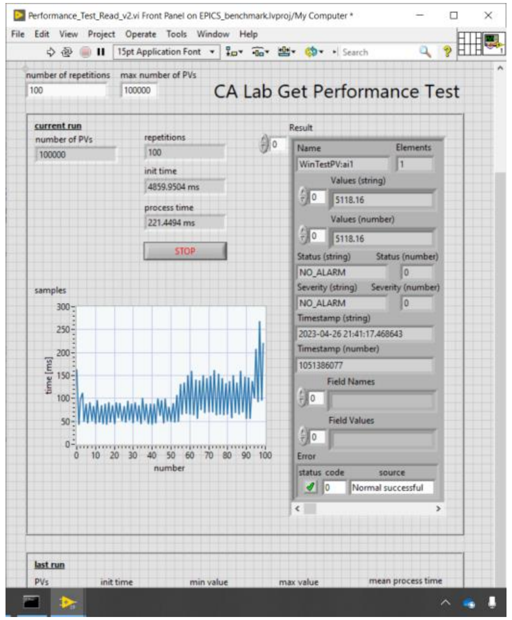
- Benchmarks of LabIOC and CALab in cooperation with Helmholtz Zentrum Berlin
- Data lost during cyberattack
- Read test on 100000 PVs, < 100 ms with both libraries



LabIOC_read_test.vi Front Panel on EPICS_benchmark.lvproj/My Compu...

number of PVs: 100000
repeats: 100
init time: 22133.6
process time: 63.7248
mean process time: 65.9776
min: 61.062
max: 80.3121
Error Out
status: ●
code: 0
source:

EPICS_benchmark.lvproj/My Computer



Performance_Test_Read_v2.vi Front Panel on EPICS_benchmark.lvproj/My Computer *

number of repetitions: 100
max number of PVs: 100000

CA Lab Get Performance Test

current run
number of PVs: 100000
repetitions: 100
init time: 4859.9504 ms
process time: 221.4494 ms

Result: 0
Name: WinTestPV:ai1
Elements: 1
Values (string): 0 5118.16
Values (number): 0 5118.16
Status (string): NO_ALARM
Status (number): 0
Severity (string): NO_ALARM
Severity (number): 0
Timestamp (string): 2023-04-26 21:41:17.468643
Timestamp (number): 1051386077
Field Names: 0
Field Values: 0
Error
status: ✓ code: 0 source: Normal successful

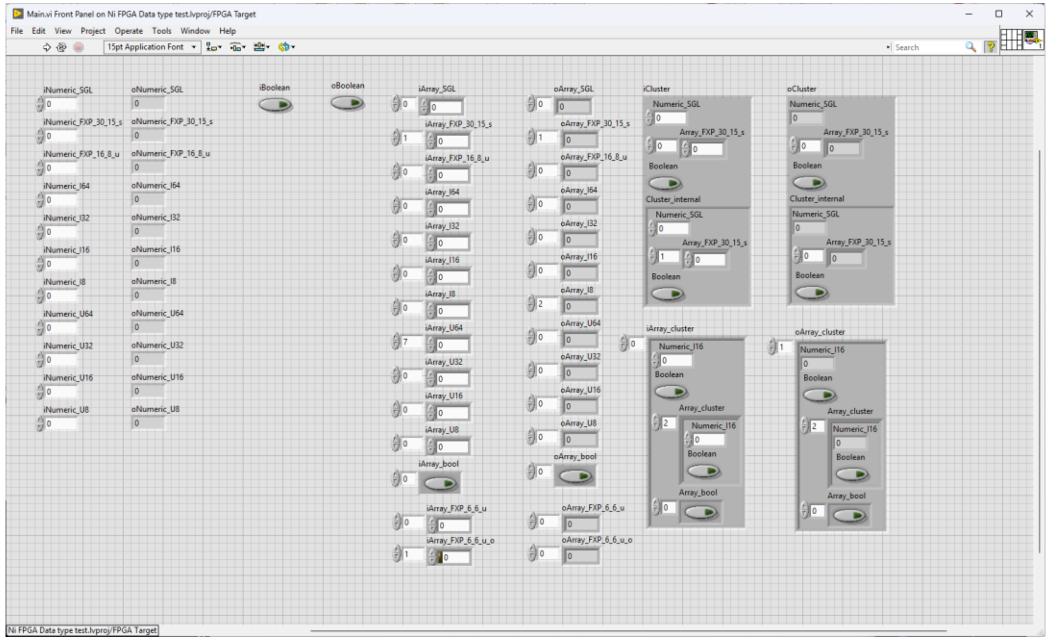
last run
PVs init time min value max value mean process time

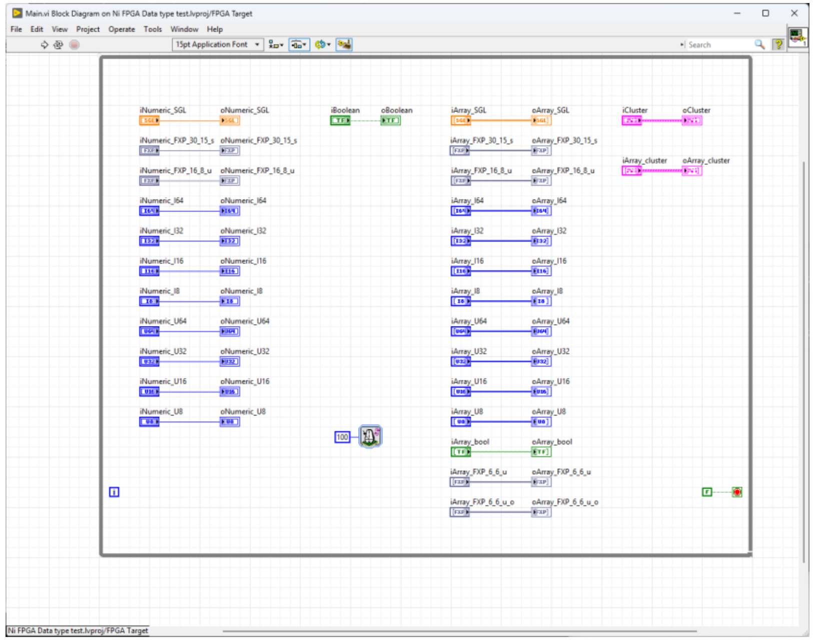
NI FPGA EPICS module

- (2023) Shift from LabVIEW to native EPICS
  - Unclear licensing policy, scaling issues, problems in CS architecture
  - Adoption of easy-to-use modules (streamdevice, Modbus, OPCUA) and Python interfaces

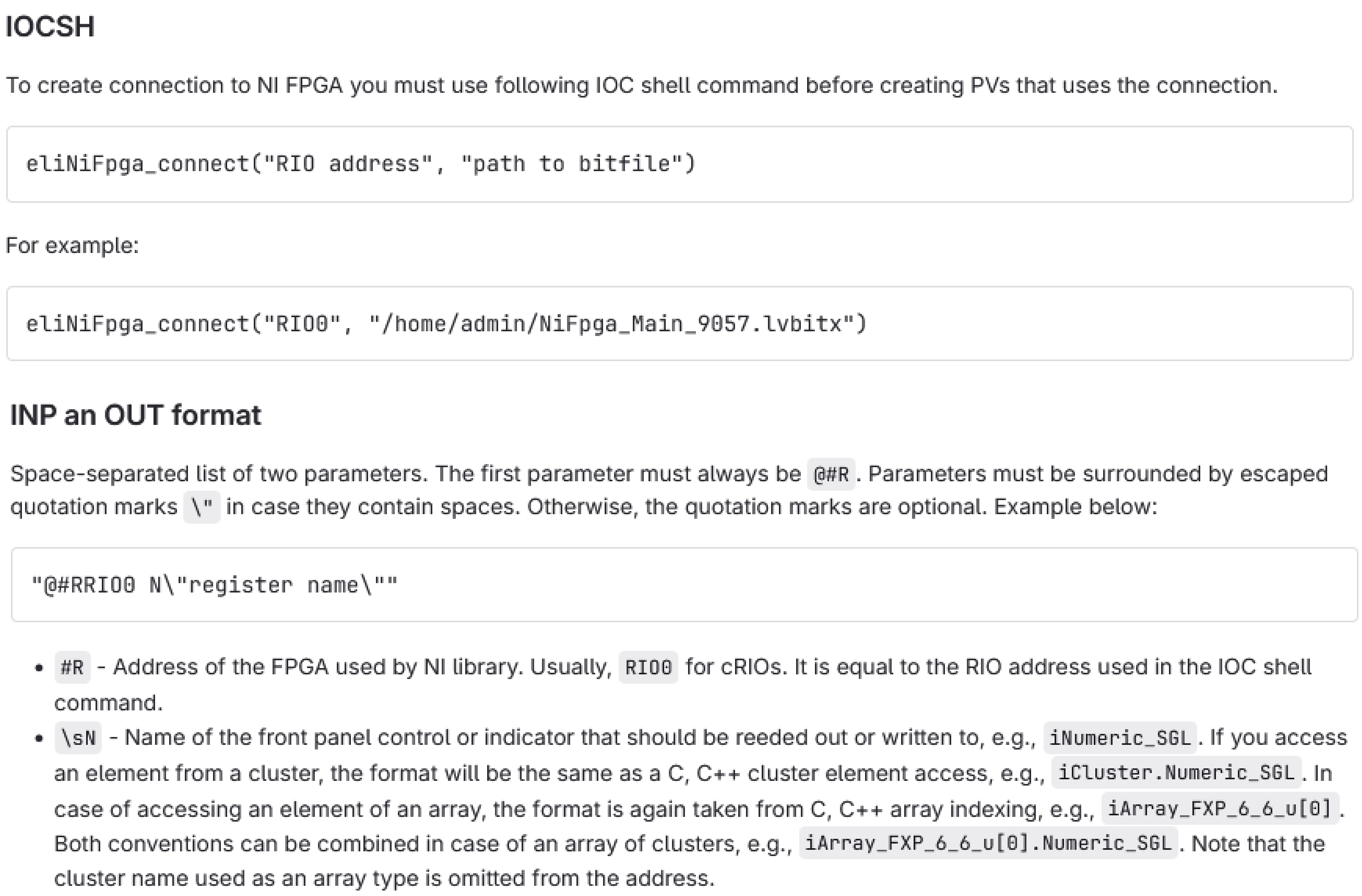
- How to interface NI FPGA devices without LabVIEW?

NI FPGA EPICS module





NI FPGA EPICS module



IOCSH

To create connection to NI FPGA you must use following IOC shell command before creating PVs that uses the connection.

```
eliNiFpga_connect("RIO address", "path to bitfile")
```

For example:

```
eliNiFpga_connect("RIO0", "/home/admin/NiFpga_Main_9057.lvbitx")
```

INP an OUT format

Space-separated list of two parameters. The first parameter must always be `@#R`. Parameters must be surrounded by escaped quotation marks `\"` in case they contain spaces. Otherwise, the quotation marks are optional. Example below:

```
"@#RRIO0 N\"register name\""
```

- `#R` - Address of the FPGA used by NI library. Usually, `RIO0` for cRIOs. It is equal to the RIO address used in the IOC shell command.
- `\sN` - Name of the front panel control or indicator that should be reeded out or written to, e.g., `iNumeric_SGL`. If you access an element from a cluster, the format will be the same as a C, C++ cluster element access, e.g., `iCluster.Numeric_SGL`. In case of accessing an element of an array, the format is again taken from C, C++ array indexing, e.g., `iArray_FXP_6_6_u[0]`. Both conventions can be combined in case of an array of clusters, e.g., `iArray_FXP_6_6_u[0].Numeric_SGL`. Note that the cluster name used as an array type is omitted from the address.

Thank you
for your attention!

Karel Majer
karel.majer@eli-beams.eu