Big Physics Day for IN2P3

16/10/2017

NI CONFIDENTIAL

NATIONAL INSTRUMENTS™

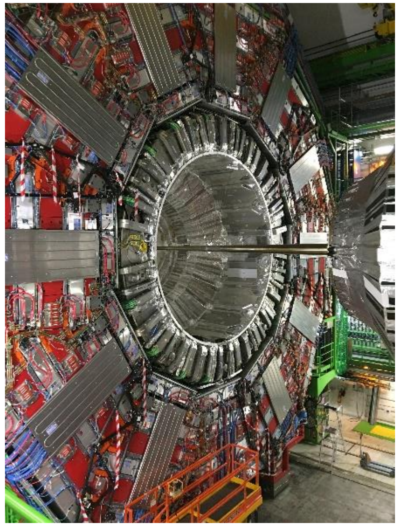
Control, Diagnostic and Measurement for Physics Systems and Experiments

Agenda

- CERN (LHC, XBOX)
- ITER Fast ICS
- ETH supercomputer
- IPP plasma experiments diagnostic
- ITER neutron detectors
- E-ELT

CERN – Usage of NI within the facility

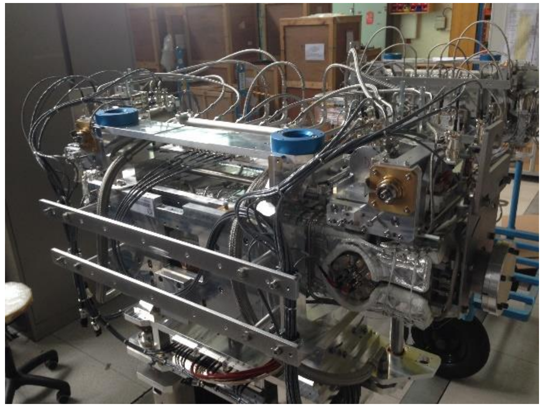
- Number of LabVIEW users on site - +500
- CERN staffed support team - 16
- Regular training class held - 10 Classes
- LabVIEW proficiency (onsite) - CLAD, CLD, CLA



CMS Detector

1 Collimator Control Systems

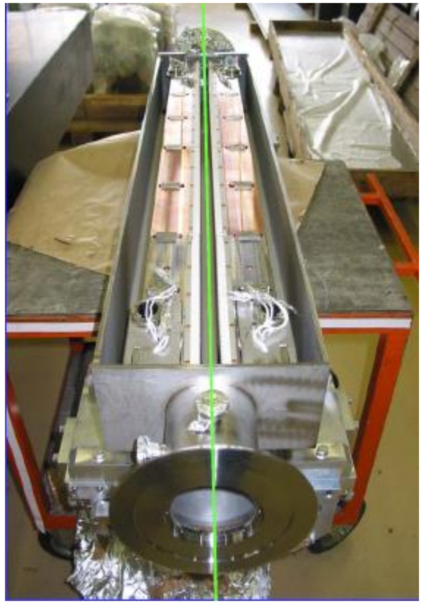
- **Some projects:**
- Collimators :- https://youtu.be/MjHaIs9hDz0
- Xbox [1,2,3] – Klystron controllers
- Magnet Safety Systems
- ….

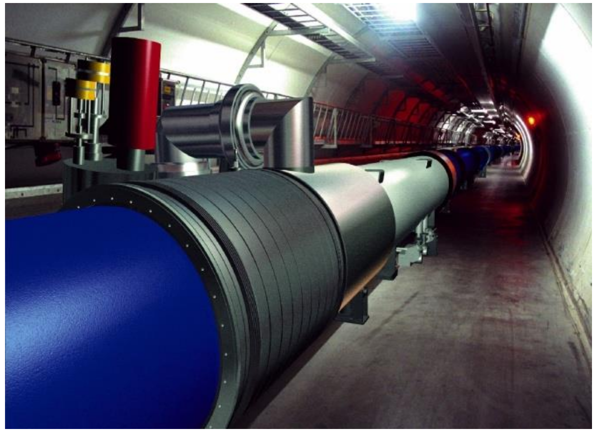


Collimator Test Bench

CERN Collimator Alignment

- 550+ axes of motion
- Across 27 km distance
- 200+ PXI systems
- The jaws have to be positioned with an accuracy which is a fraction of the beam size (200µm) with SoftMotion
- Synchronized to
    - < 5ms drift over 15 minutes
    - Maximum jitter in µs
    - 1ms of accuracy over 27 kms
- NI FPGA based hardware
- LabVIEW Real-Time
- Installed operation since 2007





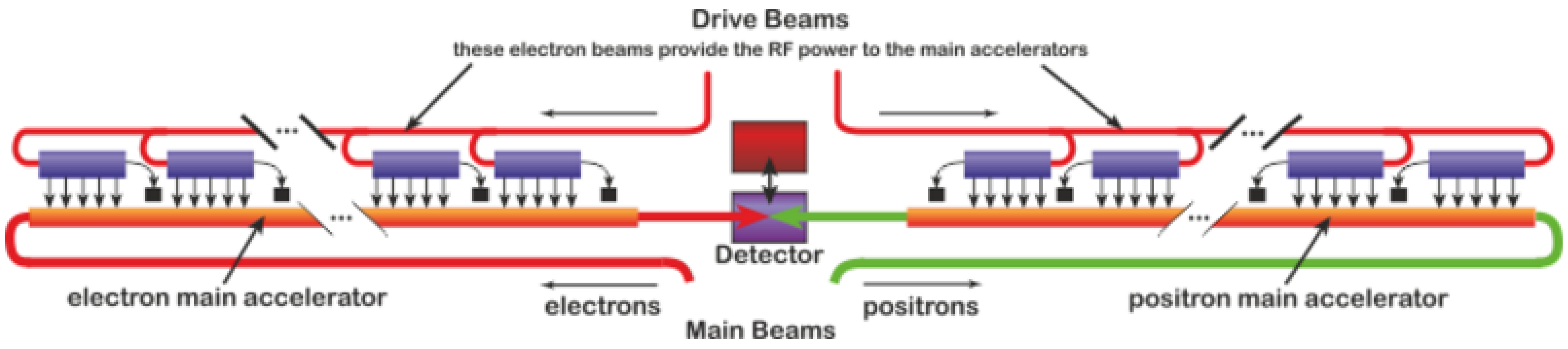
http://www.ni.com/video/598/en/

NATIONAL INSTRUMENTS™

What is CLIC?

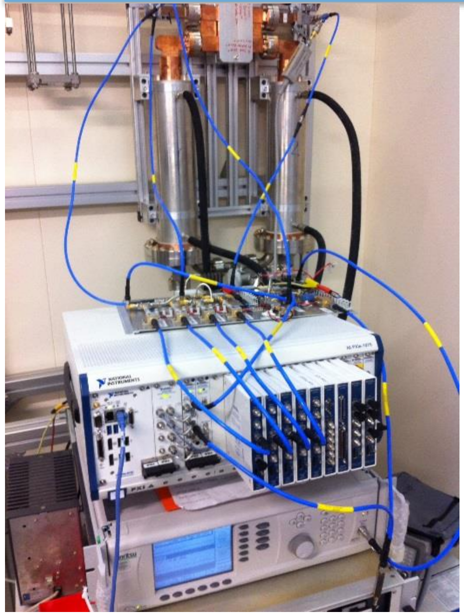
- Compact LInear Collider is a study for a future electron-positron collider that would allow physicists to explore a new energy region beyond the capabilities of today's particle accelerators thanks to electron (and not proton) collisions.

- Properties:
  - 3 TeV
  - High gradient accelerating cavities (100 MV/m, 20x LHC)
  - Operates in X band (12 GHz) to produce accelerating fields
  - Rep rate of 50 Hz



CERN – XBOX Control and Test System

Experimental physicists in the Beams RF group at CERN teamed up with NI over the past three years to develop 3 generations of high-gradient accelerating cavities conditioning and testing systems.



Project has received worldwide attention and system **duplications** were ordered by:

- SLAC
- Uppsala University
- Uni Valencia

Hardware

- PXIe-1075, PXIe-8135
- FlexRIO 5761R, 5772R, 6583R, 5793R

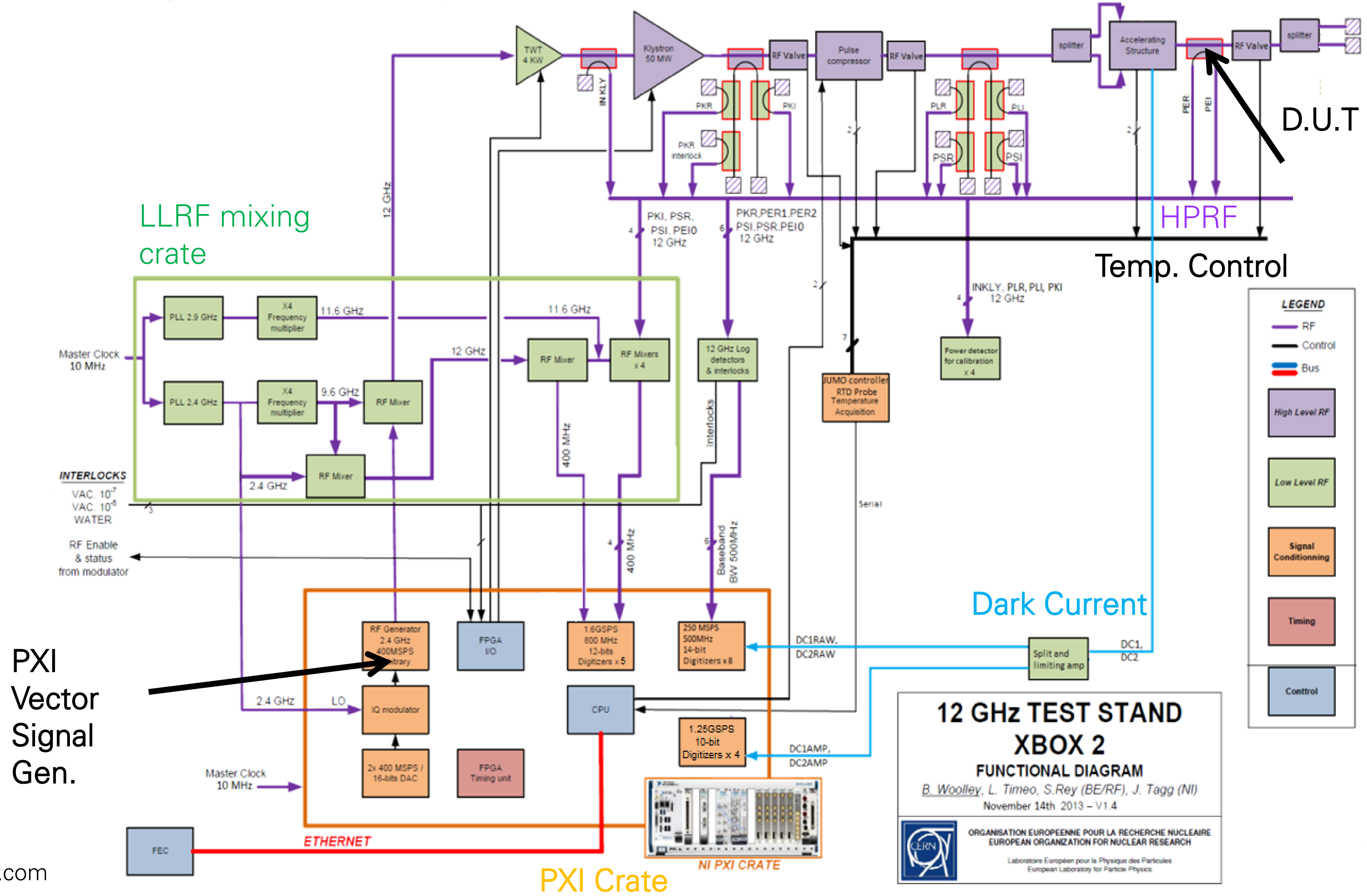
XBOX 1 - PXIe control with mixture of NI and external instrumentation
XBOX 2 - Fully PXIe-based control and instrumentation
XBOX 3 - Same as 2, but can test multiple structures simultaneously

Results

XBOX 1 & 2 are fully functional and have delivered thousands of hours worth of data. XBOX 3 is currently being assembled and tested.

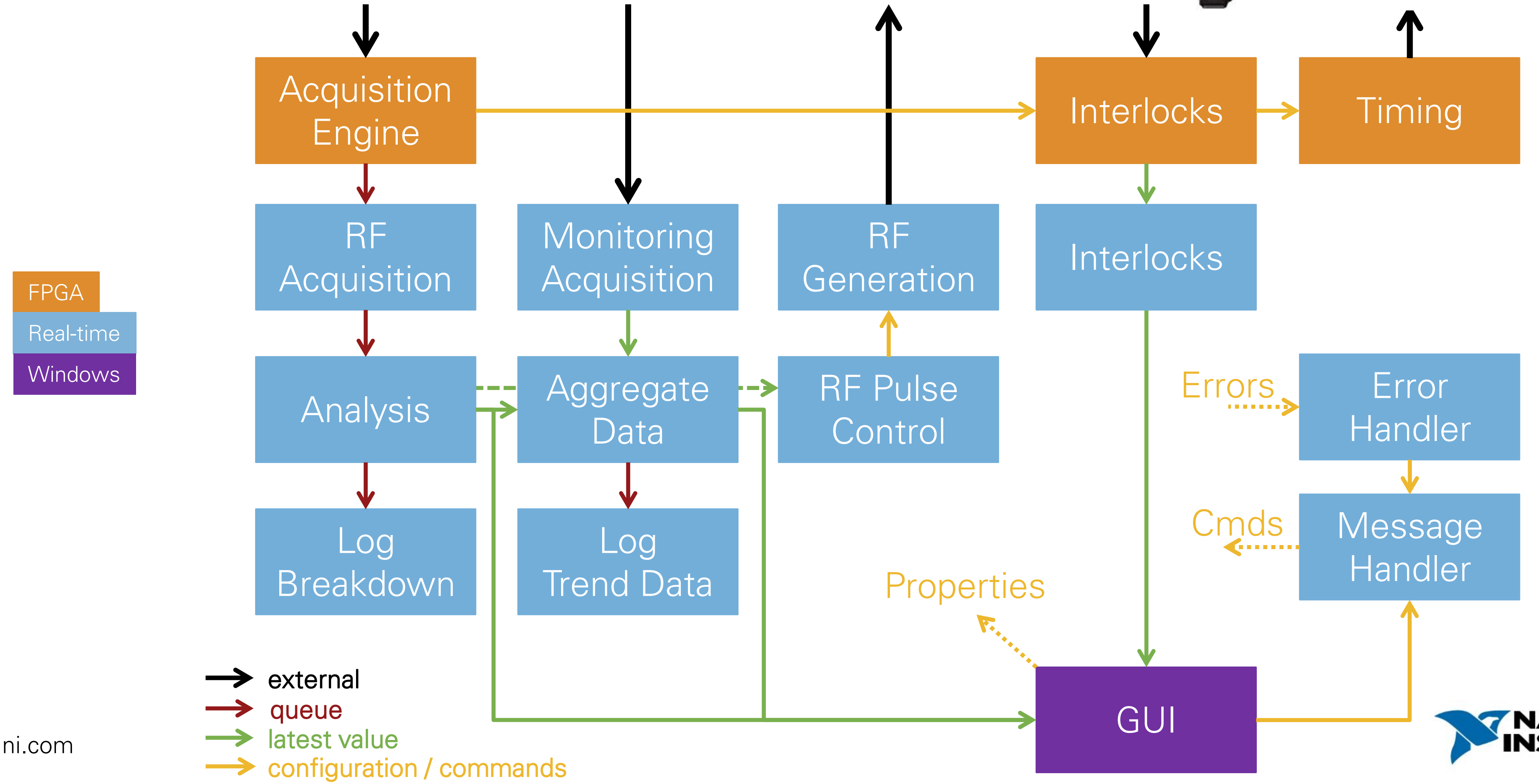
The collaboration has allowed the CLIC project to gather results quickly and in a flexible way by being able to reprogram the system as required by the project.

System Layout and diagnostics

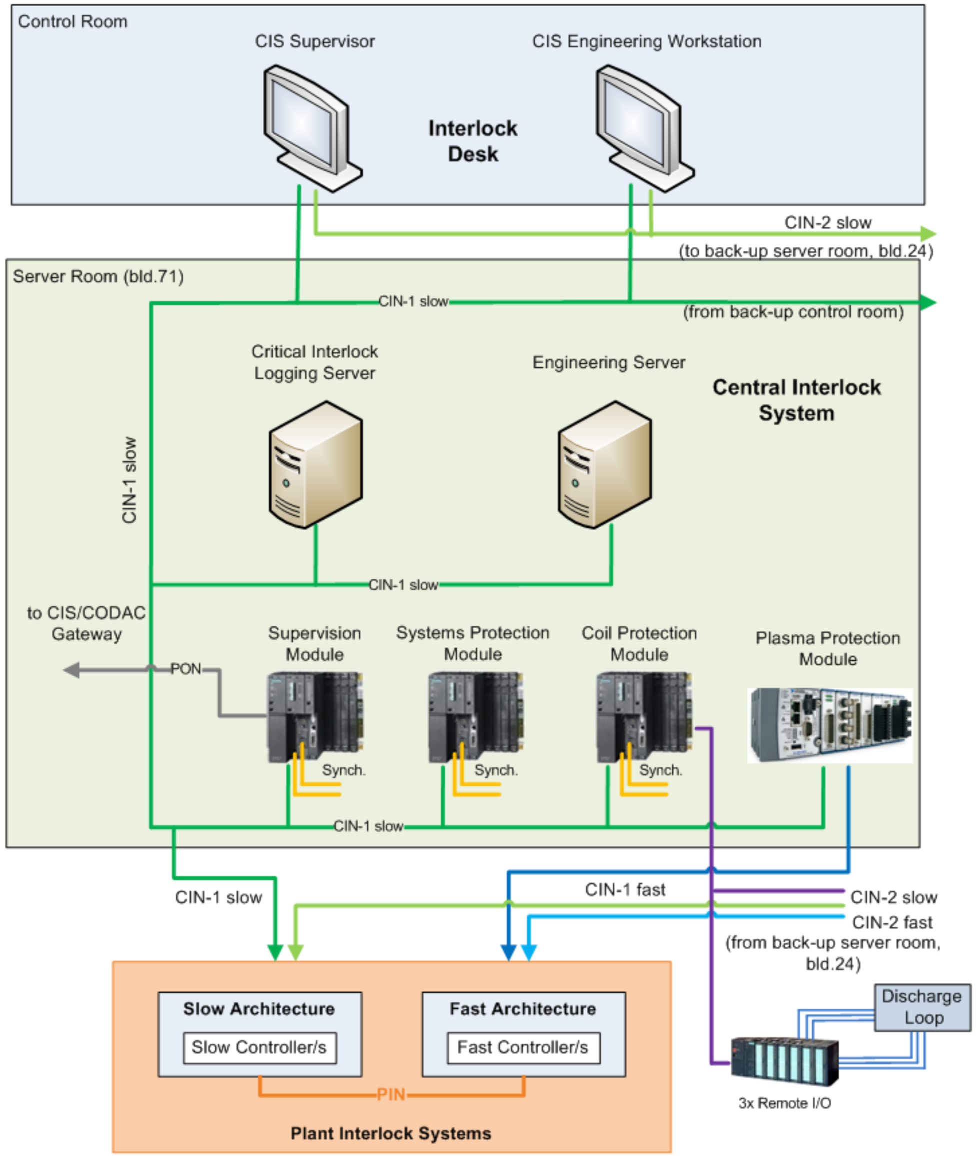


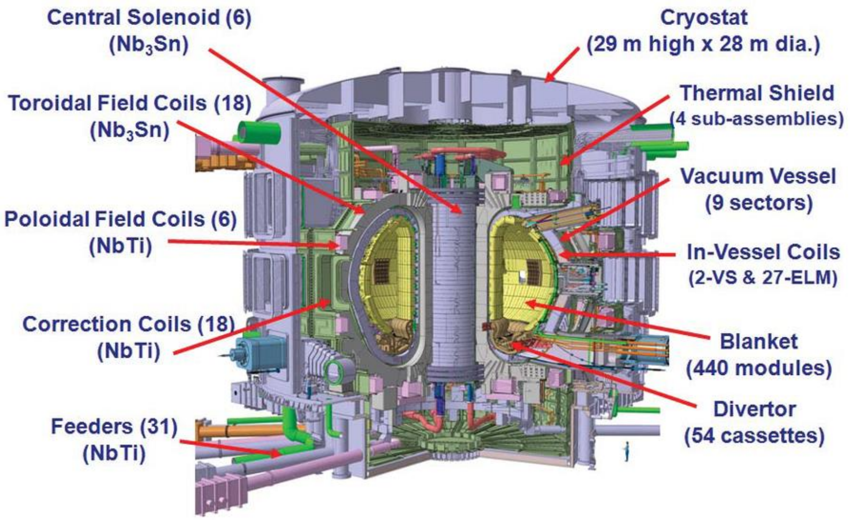
NATIONAL INSTRUMENTS

XBOX 2 software Architecture



ITER Interlock Systems



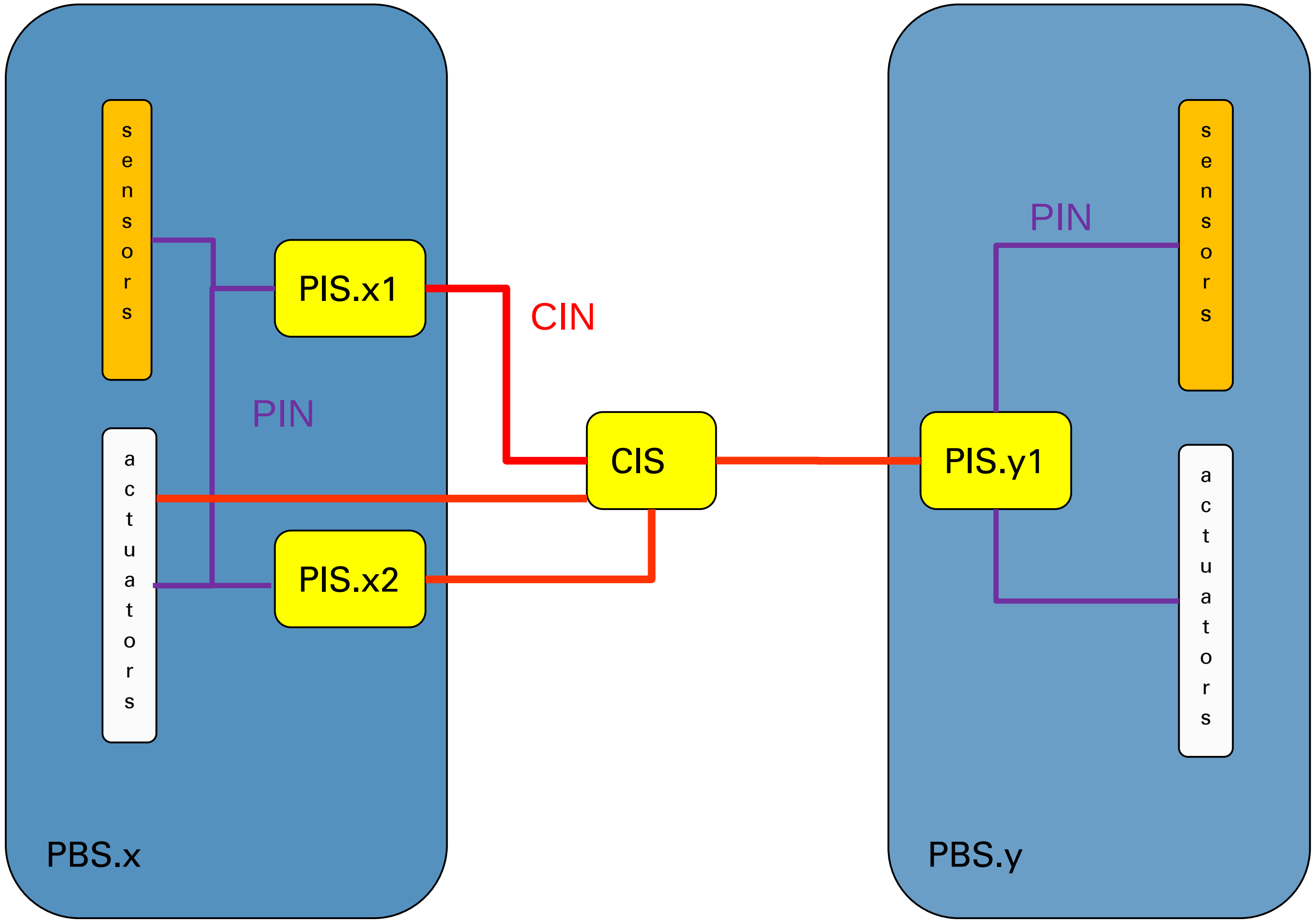


- ITER Interlocks are implemented at :

- a plant system level by the Plant Interlock Systems or PIS
- a central level by the Central Interlock System or CIS

- Slow interlocks are based on Siemens PLC
- Fast interlocks are based on FPGA redundant technologies.
- Highly critical interlocks are based on hardwired current loops

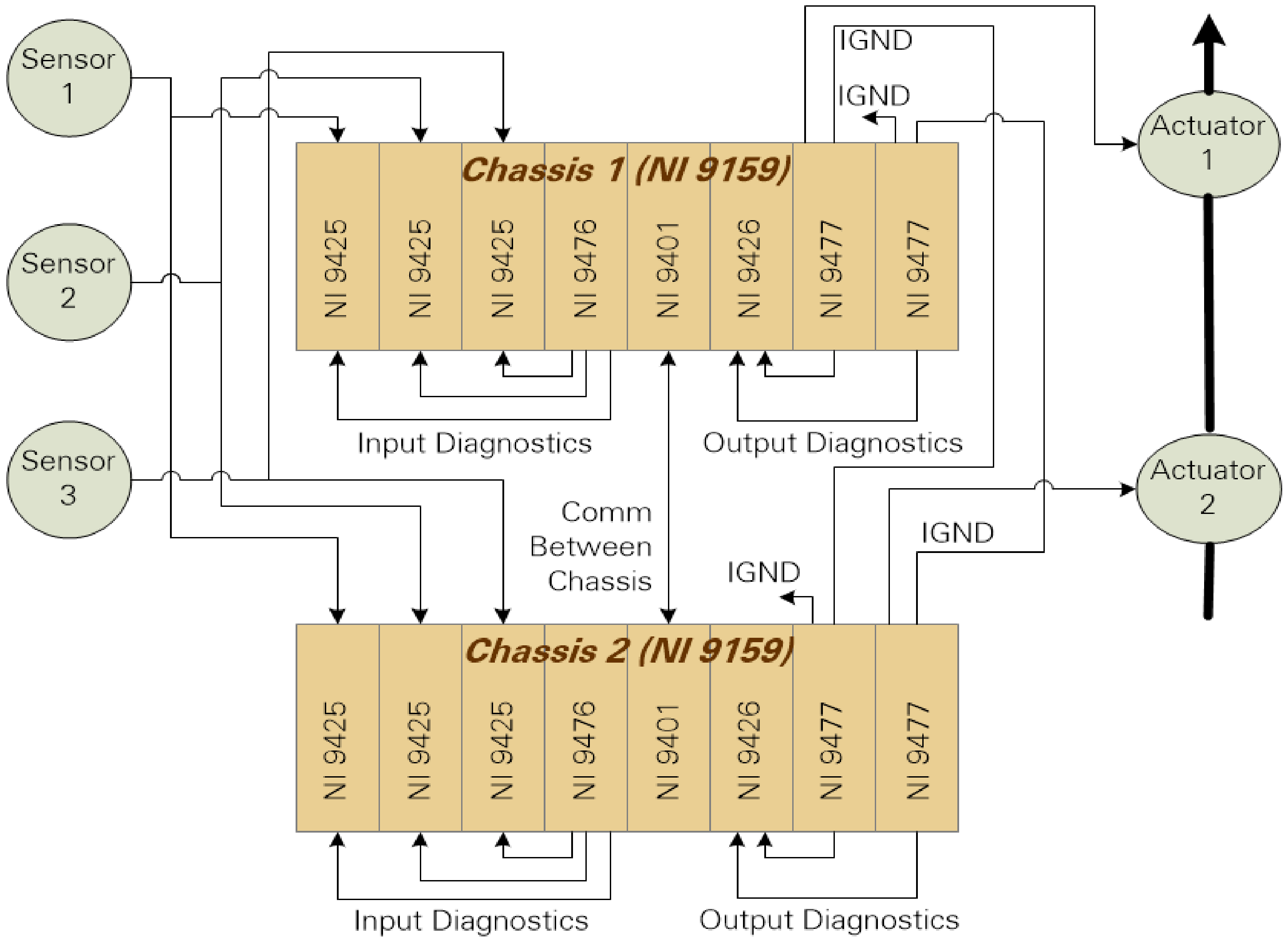
FAST interlock architecture



PIS: Plant Interlock System
CIS: Central Interlock System

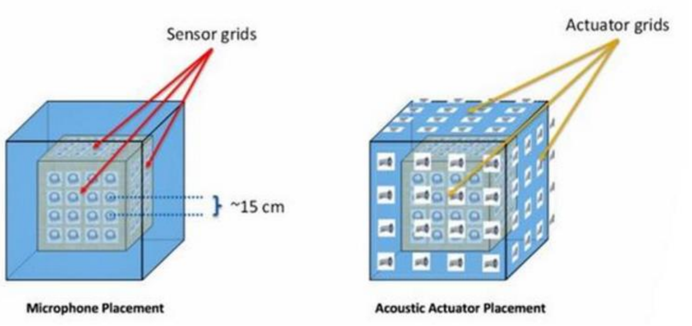
Fast interlock with cRIO

- Availability > 99.9%
- Reliability > 99.6%
- HFT (HW fault tolerance) = 1
- SFF (safe failure fraction) = 85%
- PFH (Prob. of dangerous failure/h) = 3.3 E-08
- *SIL 2 type numbers (NOT CERTIFIED)*



ETH Zurich – Development of a RealTime Numerical Simulator

Large wavelengths seismic waves extrapolation

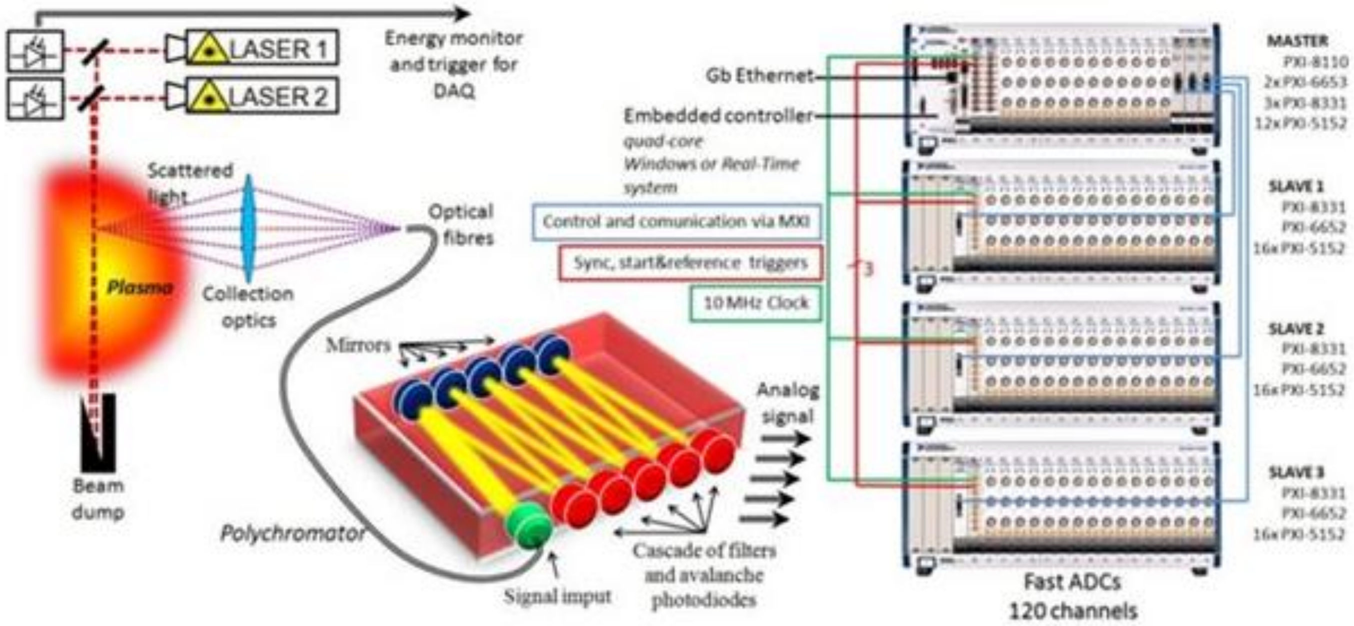


$$p^{emt}(x^{emt}, l, m) = p^{emt}(x^{emt}, l, m-1) + \sum_{x^{rec}} \left\{ \begin{array}{l} G(x^{emt}, l-m; x^{rec}, 0)\partial_j p(x^{rec}, m) - \\ \partial_j G(x^{emt}, l-m; x^{rec}, 0) p(x^{rec}, m) \end{array} \right\} n_j$$

- Acquisition: 800+ channels
- Computation: Large PDE solver
- Control: 800+ channels
- Real-time constraint: 50 µs cycle time
- More than 500 FPGA modules
- Includes a complete acquire
    - compute
    - control cycle

http://www.ni.com/niweek/keynote-videos/

Czech Institute of Plasma Physics

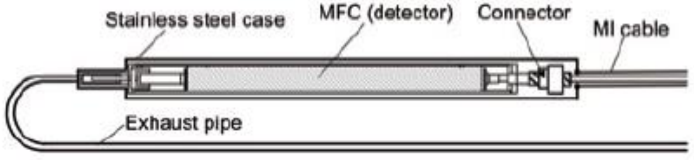


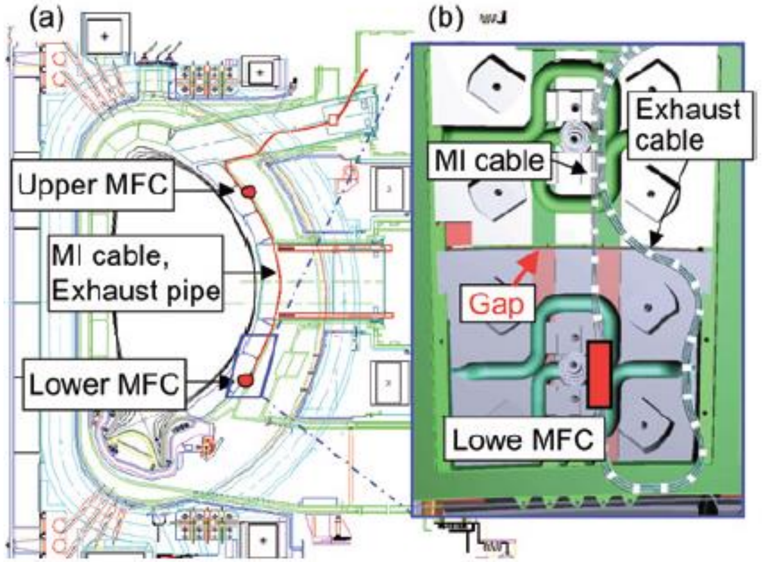
- Thomson scattering system for plasma temperature and density diagnostic
- Synchronized high speed data acquisition
  - 120 channels running at 1GS/s
  - Tight synchronization over 4 PXI chassis
  - Skew < 500 ps
  - 30Hz repetition rate

http://sine.ni.com/cs/app/doc/p/id/cs-13319#

ITER Neutron detectors

- ITER has established the use of 4 FC units, each having 3 individual detectors



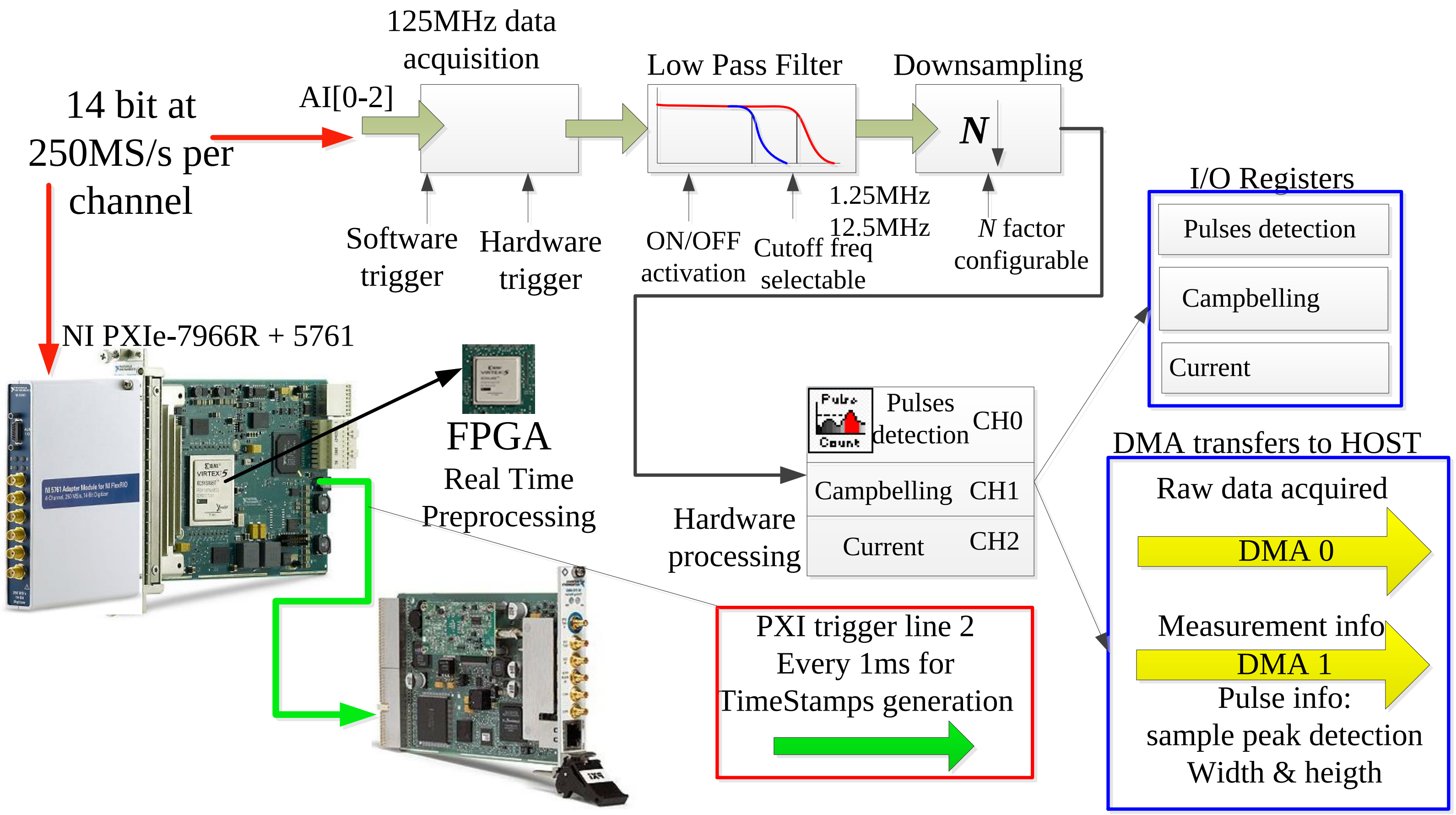


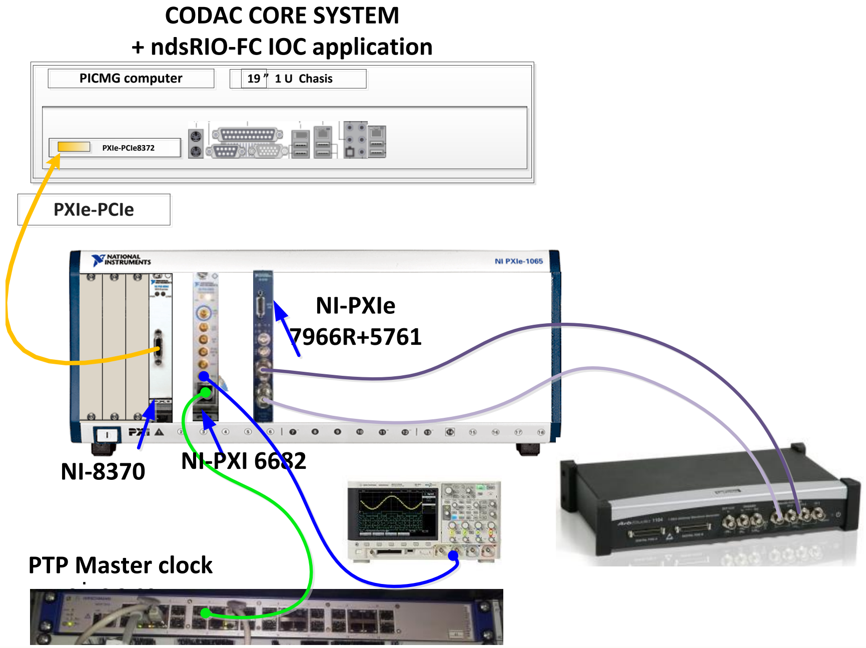
- ITER
  - Magnetic confinement of plasma by control system (CODAC)
- Diagnostics
  - Provides information to the Plasma Control System, and post-pulse analysis
  - Fusion power measurement is one of them.
  - A fraction of the fusion reaction generates neutrons.
  - Fission Chamber (FC) is based on the neutron diagnostics (temporal resolution restricted to 1 ms for counting, campbelling, and current measurements)

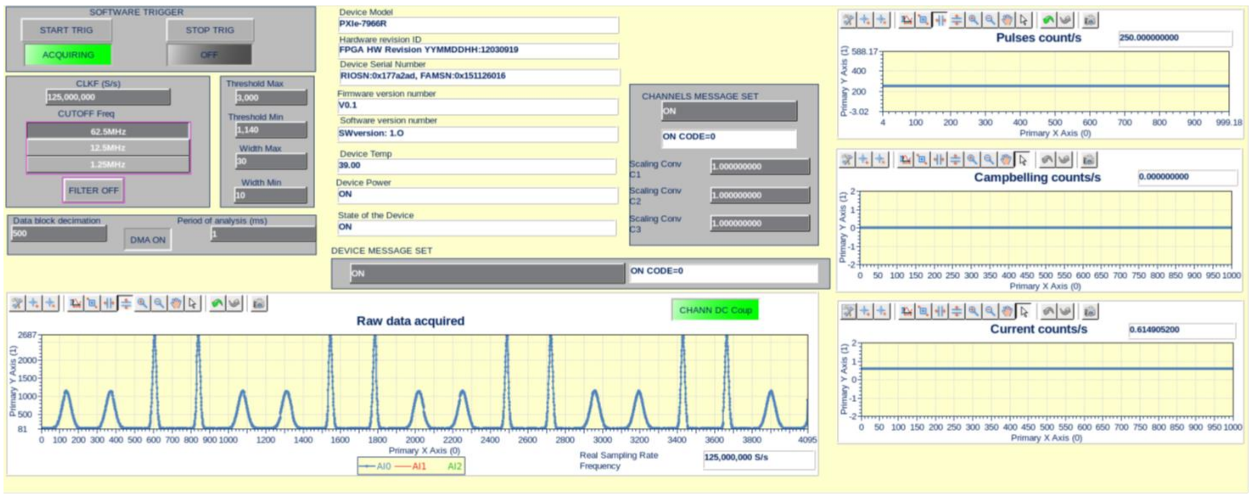
Figures taken from (M. ISHIKAWA, et al., 2008)

Hardware Implementation

- Fission chamber Measurements require Real-Time preprocessing
- Signal processing and measurements are performed by the FPGA embedded in the FlexRIO device. Timestamps provided by IEEE1588-2008
- EPICS is the high level control system

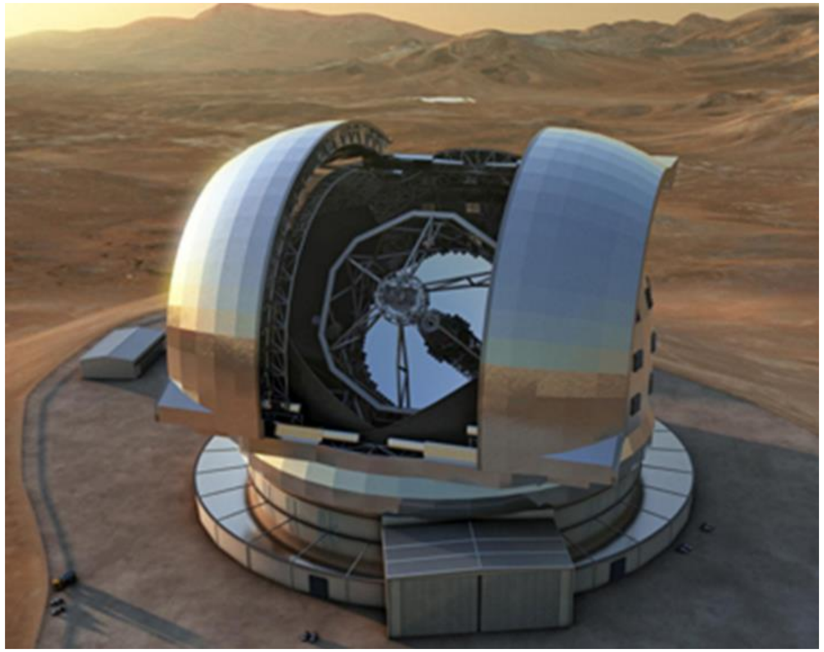






E-ELT

E-ELT is an adaptive telescope :
O control system far more complex than previous generations of telescopes.
O substantial increase of I/O points,
O higher computational and communication demands
O stronger coupling among subsystems.



- LV RT to control 800 segments positions (3000 actuators and 6000 sensors) at a rate between 500Hz et 1000kHz.
- LV RT used for HIL : Validate the NI control system with simulated plant (mirror)

http://sine.ni.com/cs/app/doc/p/id/cs-11565#

NATIONAL INSTRUMENTS

How to interface NI products to EPICS

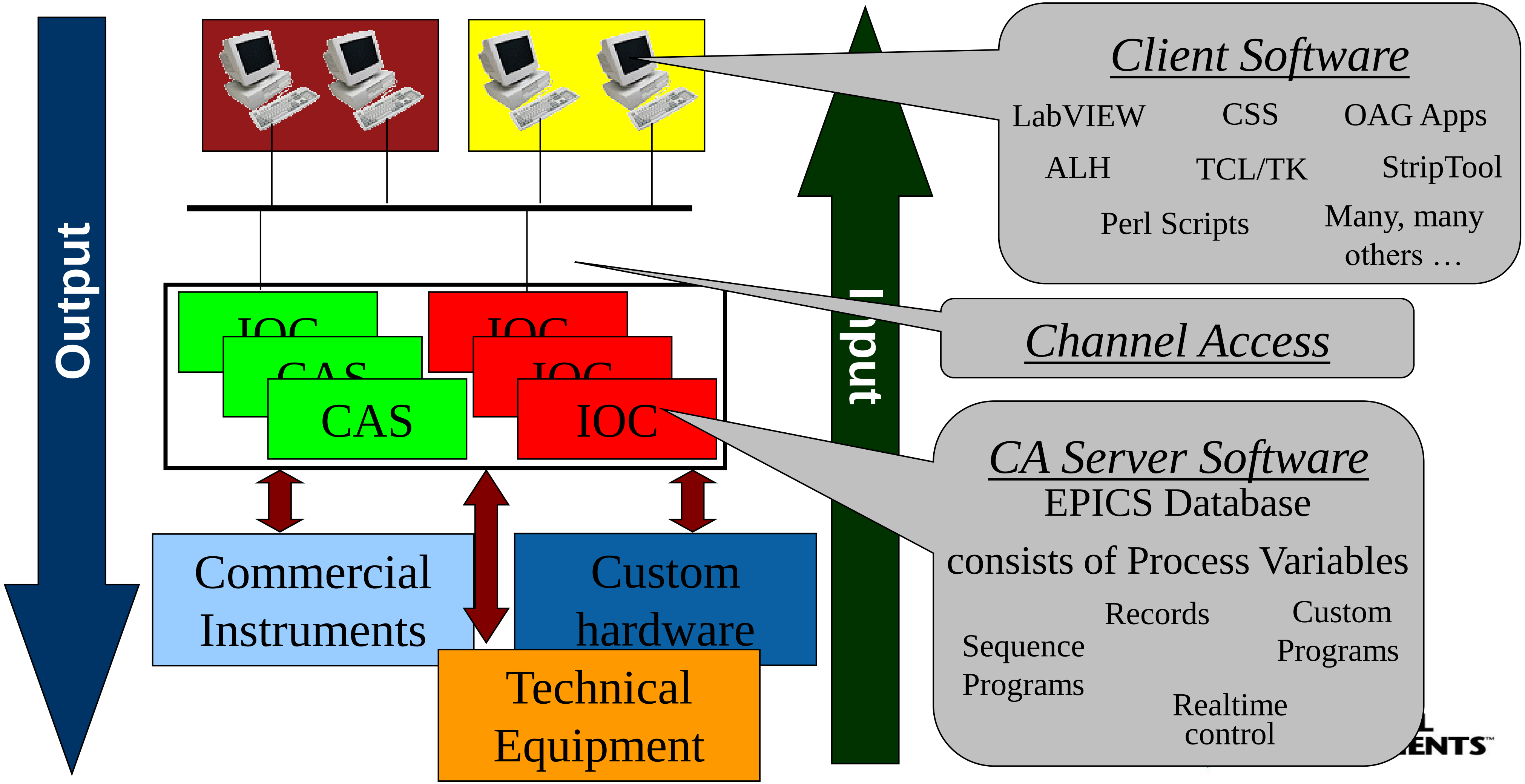
Agenda

- Main EPICS concepts
- NI-EPICS interface options

EPICS architecture

Network based Client/Server control system architecture
Servers provide information and service/ Clients request information or use services

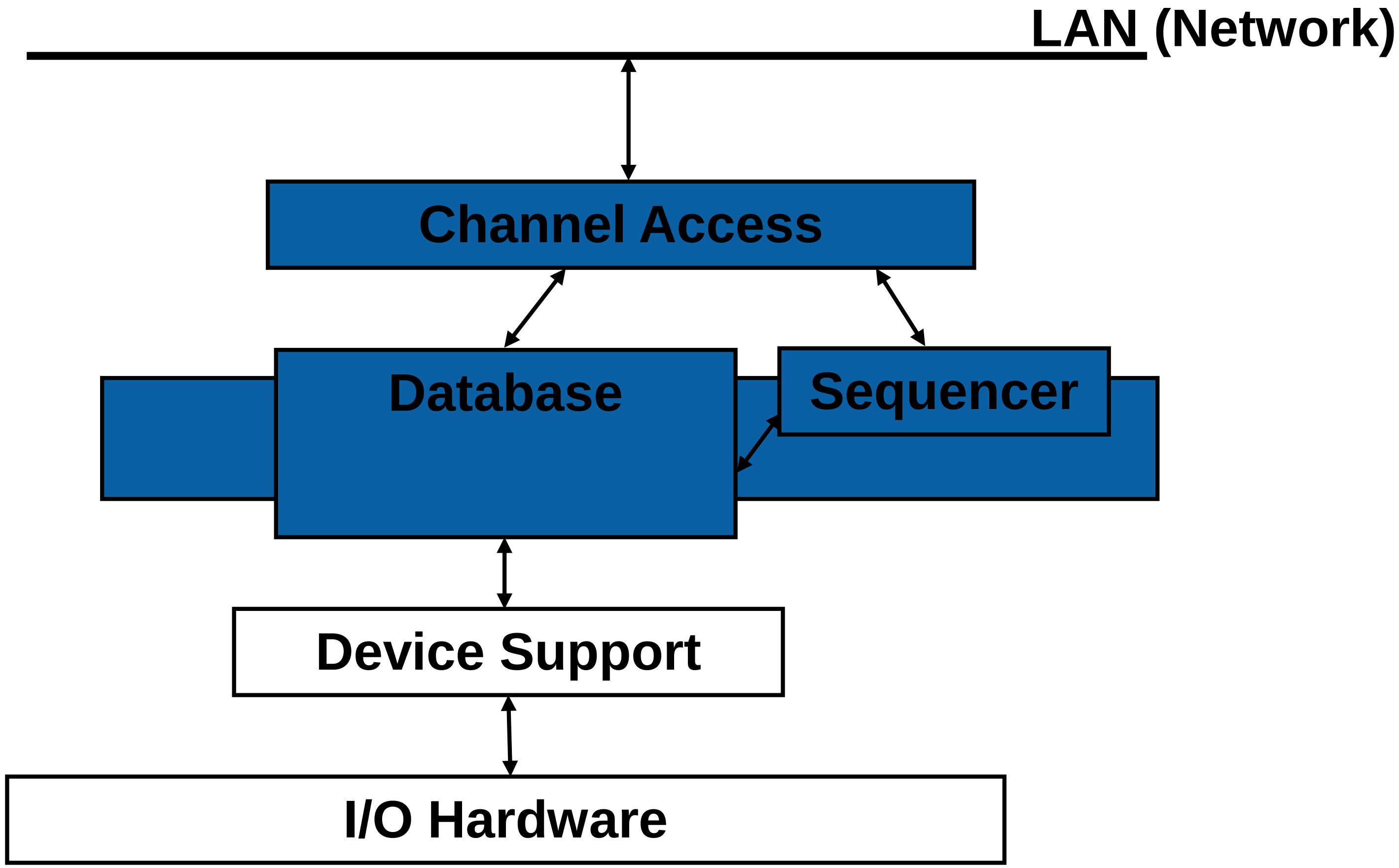
What is needed?
- An EPICS Base
- A database
- An EPICS driver
- A client



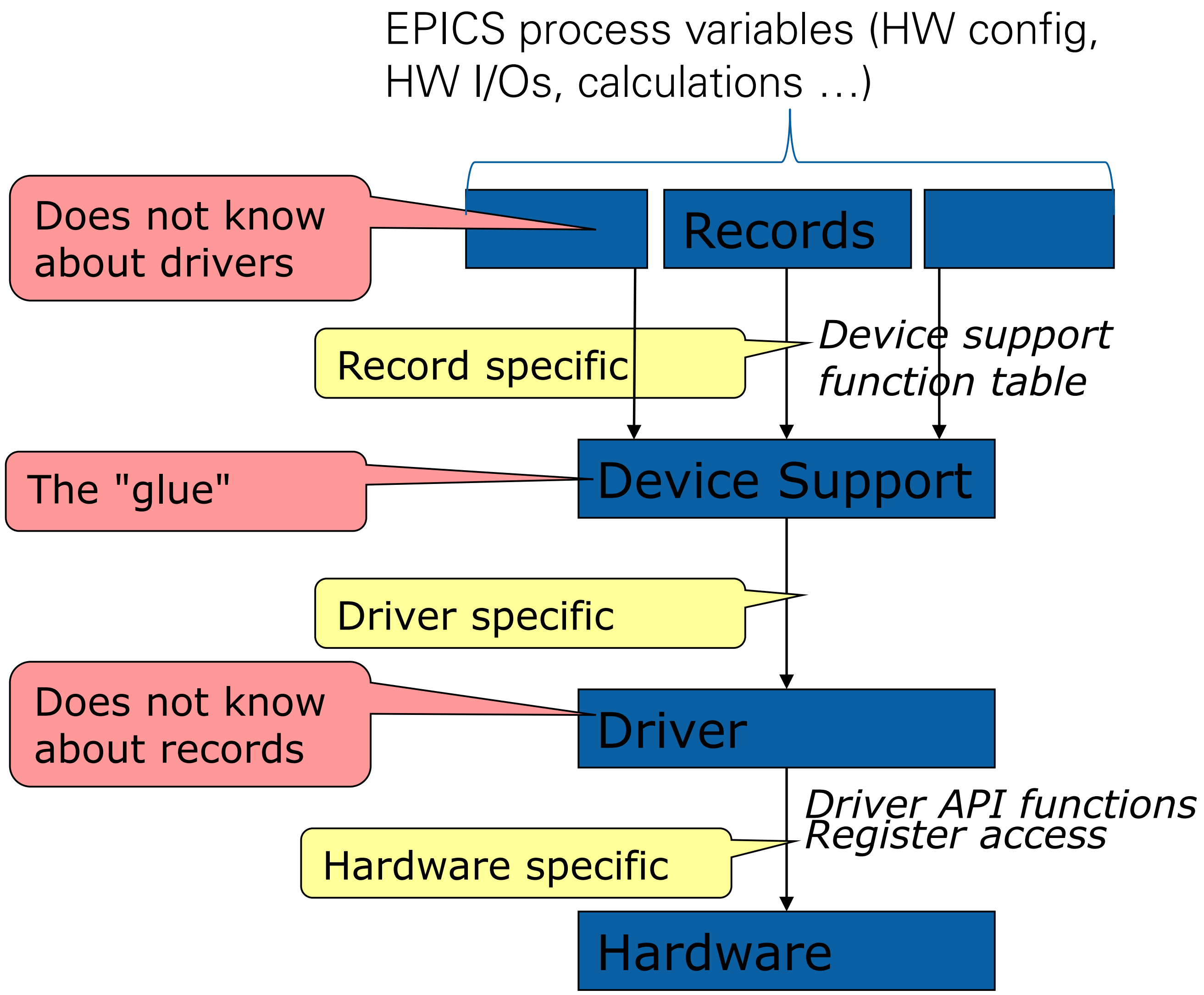
From PSI Epics Course

Inside an IOC

The major software components of an IOC



Device support and records




```
record (ai, "Room_Temperature") {
    field (EGU, "°C")
    field (LOW, "10")
    field (HIGH, "40")
    field (HOPR, "50")
    field (LOPR, "0")
    field (DESC, "Room A temp")
    field (DTYP, "NI 6268 ")
    field (INP, "#C0 S0)
    field (SCAN, "1 Second")
}
```

Analog out device support (write)

```
long myDacWriteAo (aoRecord *record)
{
  myDacAoPrivate *priv = (myDacAoPrivate*) record->dpvt;
  int status;

  if (!priv) {
    recGblSetSevr (record, UDF_ALARM, INVALID_ALARM);
    errlogSevPrintf (errlogFatal,
      "myDrvWriteAo %s: record not initialized correctly\n",
      record->name);
    return -1;
  }

  status = myDacSet (priv->card, priv->signal, record->rval);
  if (status) {
    errlogSevPrintf (errlogFatal,
      "myDrvWriteAo %s: myDacSet failed: error code 0x%x\n",
      record->name, status);
    recGblSetSevr (record, WRITE_ALARM, INVALID_ALARM);
  }
  return status;
}
```

Get private data back from dpvt

Check for proper initialization

Call driver function

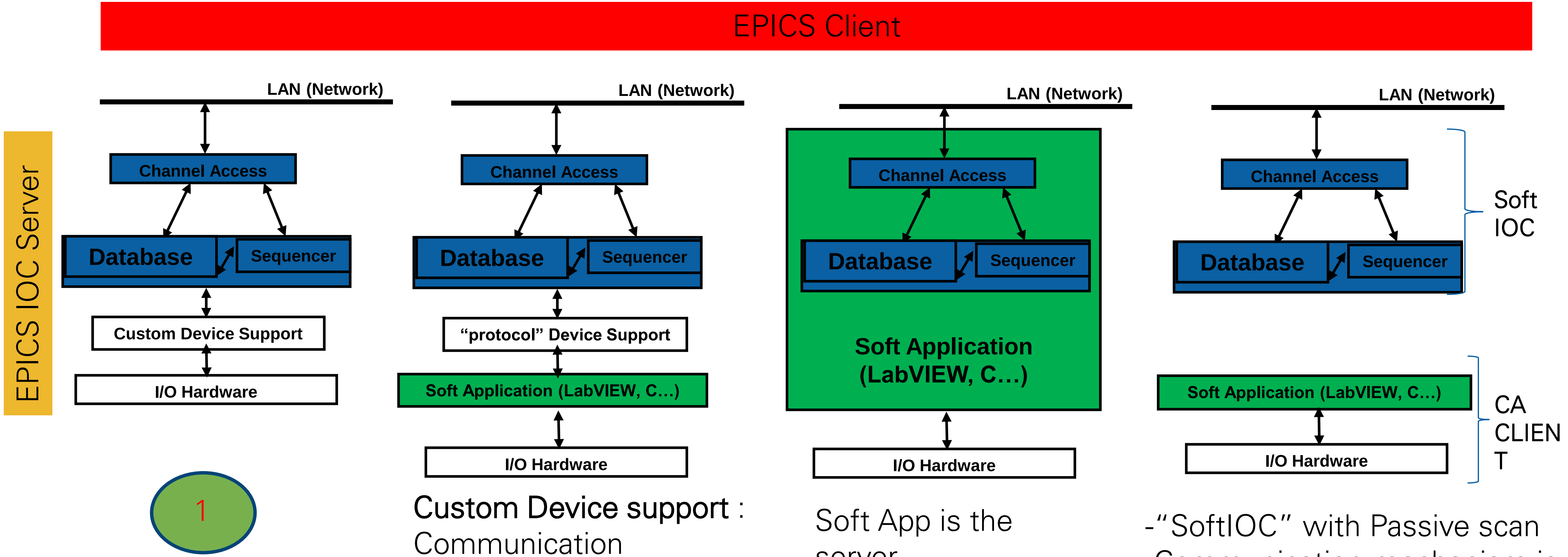
Return 0 or error status

NATIONAL INSTRUMENTS™

Records processing

- Record processing can be periodic or event driven or passive
- **Periodic**:
  - Standard scan rates: 10, 5, 2, 1, 0.5, 0.2 and 0.1 seconds
  - Custom scan rates can be configured up to speeds allowed by operating system and hardware
- **Event driven**:
  - Hardware interrupts
  - EPICS Events (post_event)
- **Passive**:
  - Channel Access Puts (caput)
  - Request from another record via links

Architectures with an intermediate software layer



Custom Device support : Communication mechanism (shared memory, protocol,..)
Records scan: Periodic, Events, Passive

2

Soft App is the server (PCAS, shared variables,etc..)

3

-“SoftIOC” with Passive scan
-Communication mechanism is actually Channel Access, so the Soft App is also an EPICS client

4

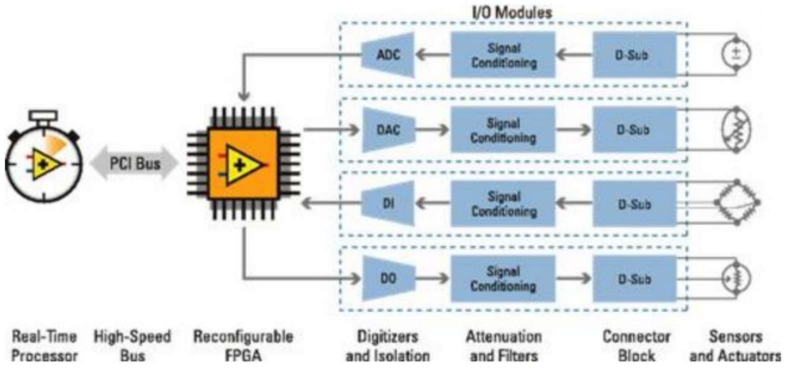
EPICS-NI interface

NI Platforms

NI
CompactRIO

- Rugged form factor with a processor and a reconfigurable FPGA
- Real-Time OS (VxWorks, Linux RT)
- Designed for harsh environments (temperature, shocks, passive cooling, etc..)
- High density hot swappable I/O modules, with built-in conditioning
- Advanced control, signal processing, modular prototyping, etc…

POWERED BY
NATIONAL INSTRUMENTS
LabVIEW

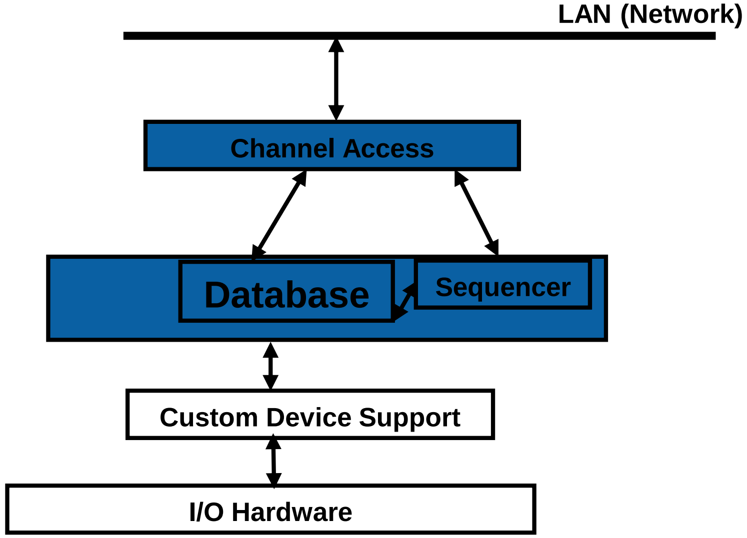


PXIe

- PCI/PCIe extended form factor with built-in timing and synchronization
- Windows/Linux/Real-Time embedded/remote controllers
- Until 24GB/s of system throughput, 8GB/s per slot, 3.6 GB/s storage speed
- More than 600 NI instruments (DAQ, digitizers, multimeters, generators, power supplies, switching, RF analyzers and generators, industrial buses…)

Embedded EPICS Base + No software layer

CSS EPICS client



NI Linux RT

Windows
Linux

Embedded EPICS Base
Server
C device support (C drivers)

```
admin@NI-cRIO-9068-BigPhysics:/home/RDS/Desktop/base-3.14.10/bin/linux-arm# ./so
ftIoc ../../cRIOApp-start.cmd
epicsEnvSet("ARCH","linux-arm)
cRIOApp-start.cmd -- Line 1 -- Unbalanced quote.
epicsEnvSet("IOC","cRIOApp")
dbLoadRecords("/home/RDS/Desktop/base-3.14.10/cRIOApp_records.db")
iocInit
Starting iocInit
############################################################################
## EPICS R3.14.10 $R3-14-10$ $2008/10/27 19:39:04$
## EPICS Base built Jan  6 2014
############################################################################
iocRun: All initialization complete
epics> dbl
CPU_load
DC
Error_code
RMS
Signal_type
Amplitude
Frequency
Offset
Monitor_meas
Monitor_signal
Save
Start
Stop
Signal_data
epics>
```

1

External EPICS Server + “communication mechanism” device support

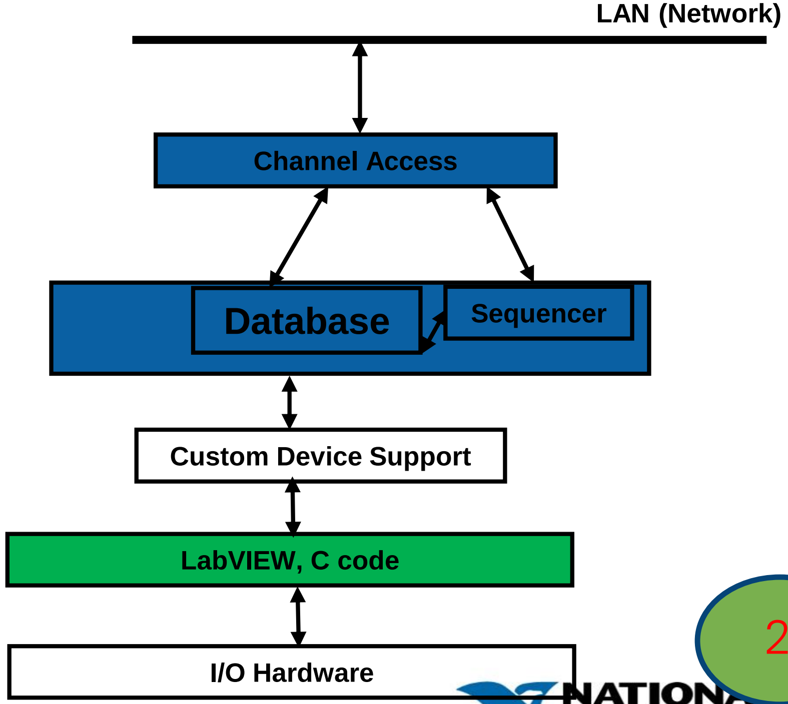
CSS EPICS client (remote control of LV App)

EPICS Server
+ Application or Network Protocol “device support”

VxWorks (LabVIEW RT), Linux RT

Pharlap (LabVIEW RT), Windows, Linux

Adapted LabVIEW or C App with a communication mechanism (TCP/IP, shared memory, etc..)



2

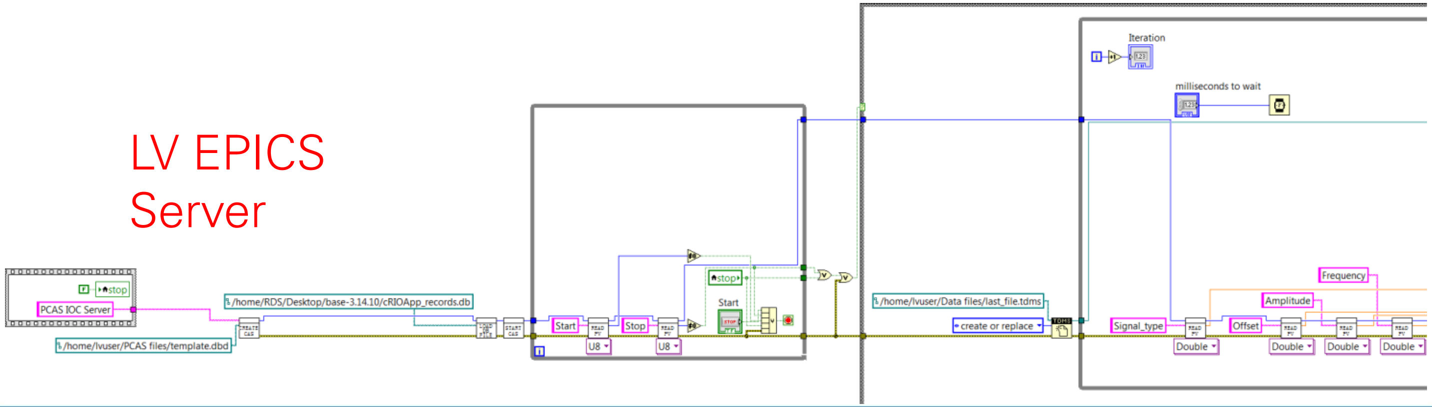
LabVIEW App as a Server

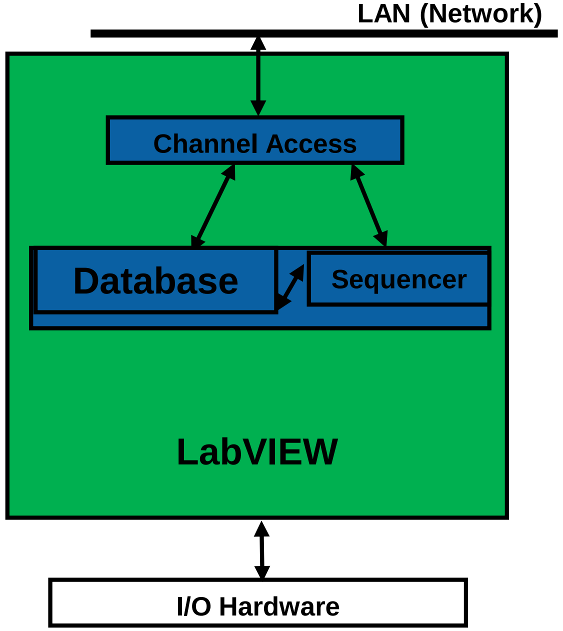
CSS EPICS client (remote control of LV RT App)

VxWorks (LabVIEW RT), NI Linux RT

Pharlap (LabVIEW RT),
Windows
Linux

LV EPICS Server



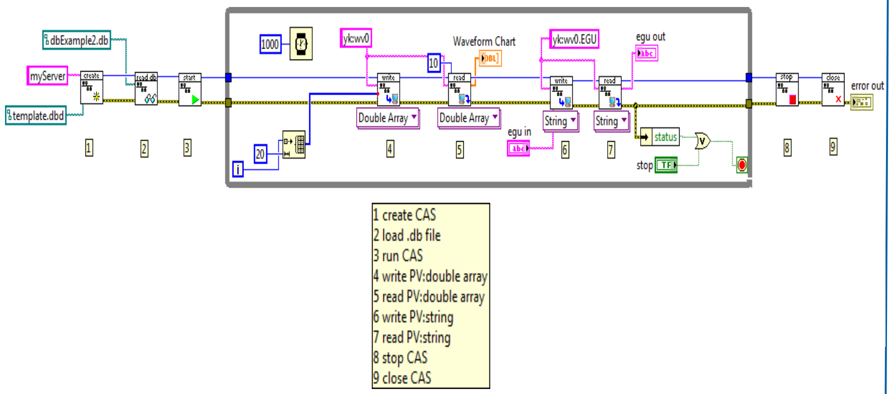


3

LabVIEW App as a Server : at least 3 options

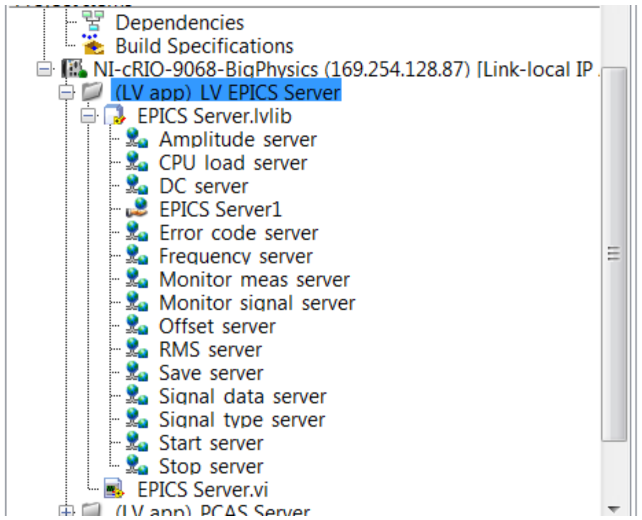
PCAS (DEMO)

- C++ class library available in EPICS base
- Ai/ao/bi/bo/waveform supported
- Needs additional development to support additional records/fields
- Requires a .dB file
- Supported on Windows, VxWorks, Linux-arm and Linux-x86



Shared variables

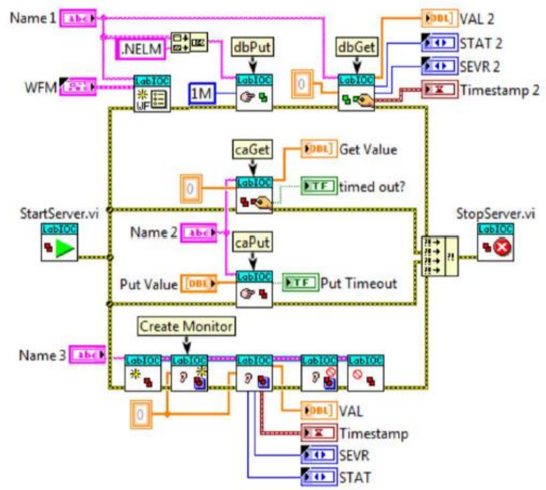
- Built-in LabVIEW RT and DSC
- No dB file
- Programmatic creation of CA Server and variables
- Only VAL field supported (alarms fields on LV DSC)



LabIOC

- Developed by the observatory of science for ELI beamlines lasers
- Full support of core EPICS records
- Relies only on native LabVIEW functions

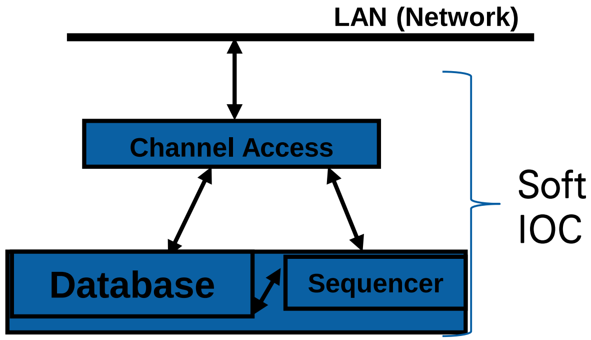
Example LabIOC VI Set



NATIONAL INSTRUMENTS

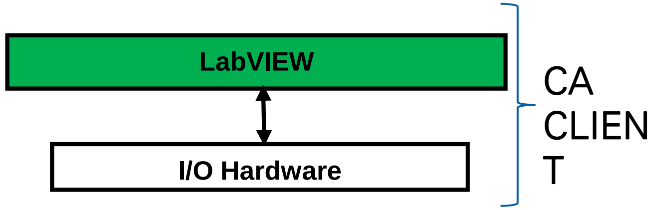
LabVIEW App as an EPICS client + Soft IOC (DEMO)

CSS EPICS client
(remote control of LV App)



NI Linux RT, VxWorks (LV RT)

Windows
Linux
Pharlap (LV RT)

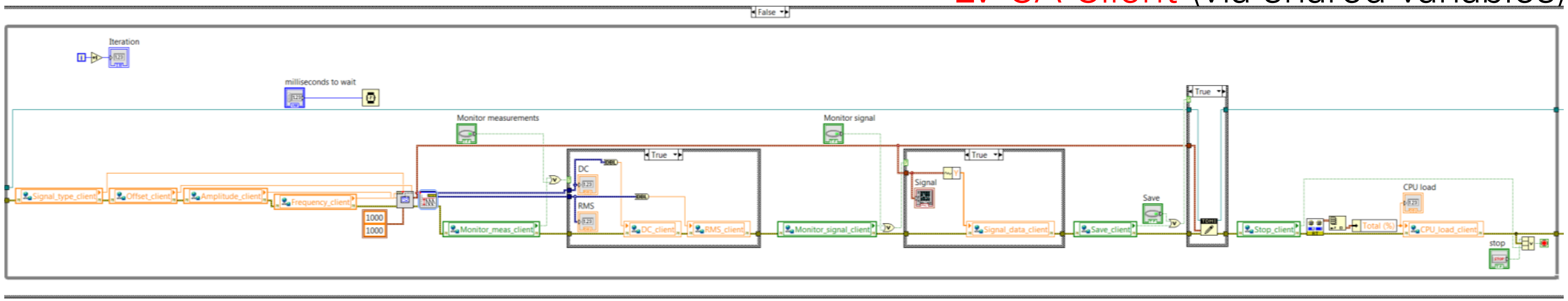


Embedded EPICS Base Server (SoftIOC)

OR

Remote

LV CA Client (via shared variables)



4

NATIONAL INSTRUMENTS

Options comparison

OPTION	LabVIEW	Device Support	EPICS Base	Pros	Cons
Embedded EPICS Server + C device support	NO	YES	YES	Full-blown EPICS Server	-Need to install EPICS Base -Need to have C drivers
External EPICS Server + « protocol » device support	YES	YES	YES	Full-blown EPICS Server	-Need to have/install an EPICS Base -need to agree on a protocol and write device support for LV App (additional software layer)
LabVIEW as an EPICS Server (PCAS or shared variables or LabIOC or CALab etc…)	YES	NO	NO	-No need to install an EPICS Base -No need to develop a device support -Minimal set-up (the app is the server)	-not flexible/complete if using shared variables -some features may have to be added if using PCAS - Some specific features could be missing if using other options
SoftIOC + LabVIEW as a client	YES	NO	YES	-No device support to write -Available shared variables to interface to PVs	-Need to install an EPICS Base -Need to use 2 clients

Conclusion

Several options to interface NI products with EPICS…

- Using LabVIEW as an IOC Server/client:
  - Built-in : shared variables
  - Add-ons by NI : PCAS VI library
  - 3rd party add-ons : Shared memory, LabIOC, CALab, etc…
- Without using LabVIEW:
  - cRIO : NI-RIO C API with NI Linux RT and Embedded Epics Base
  - PXI : custom device support with instruments C drivers

…. On several operating systems (Windows, Linux desktop, VxWorks, Pharlap, NI Linux RT)

TANGO interfacing with NI technology

- TANGO concepts
- From NI HW/ NI LabVIEW to TANGO

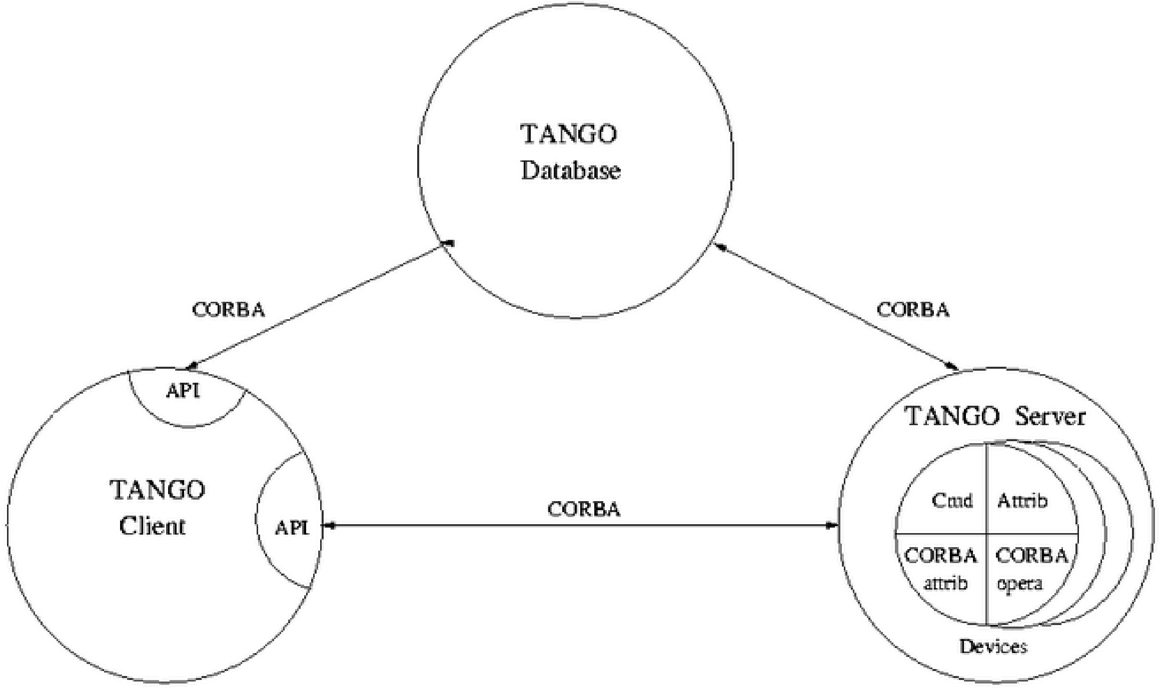
What is TANGO?

The TANGO control system is a free, open source, object oriented, distributed control system (based on CORBA) used for controlling synchrotrons, lasers, physics experiments in over 20 sites.

It was initially developed by ESRF and is now developed as a collaborative effort between Alba,Anka, Desy, Elettra, ESRF, FRM II, Solaris ,
MAX-IV and Soleil institutes.

Architecture

-It’s based on a client/server model (in C++, Java or Python).
-It uses CORBA/Zeromq for network communication and the concept of Device Classes with object oriented programming.
-Clients import these Devices via a database.



TANGO devices

TANGO software bus

Motor

Commands: On(), Off(), …

Attributes: Speed, Position

State: On, Off, Alarm, Fault

Polling, round robin buffer, threading, event triggering, archiving …

Hardware control code

Device class

Interface

Code generator

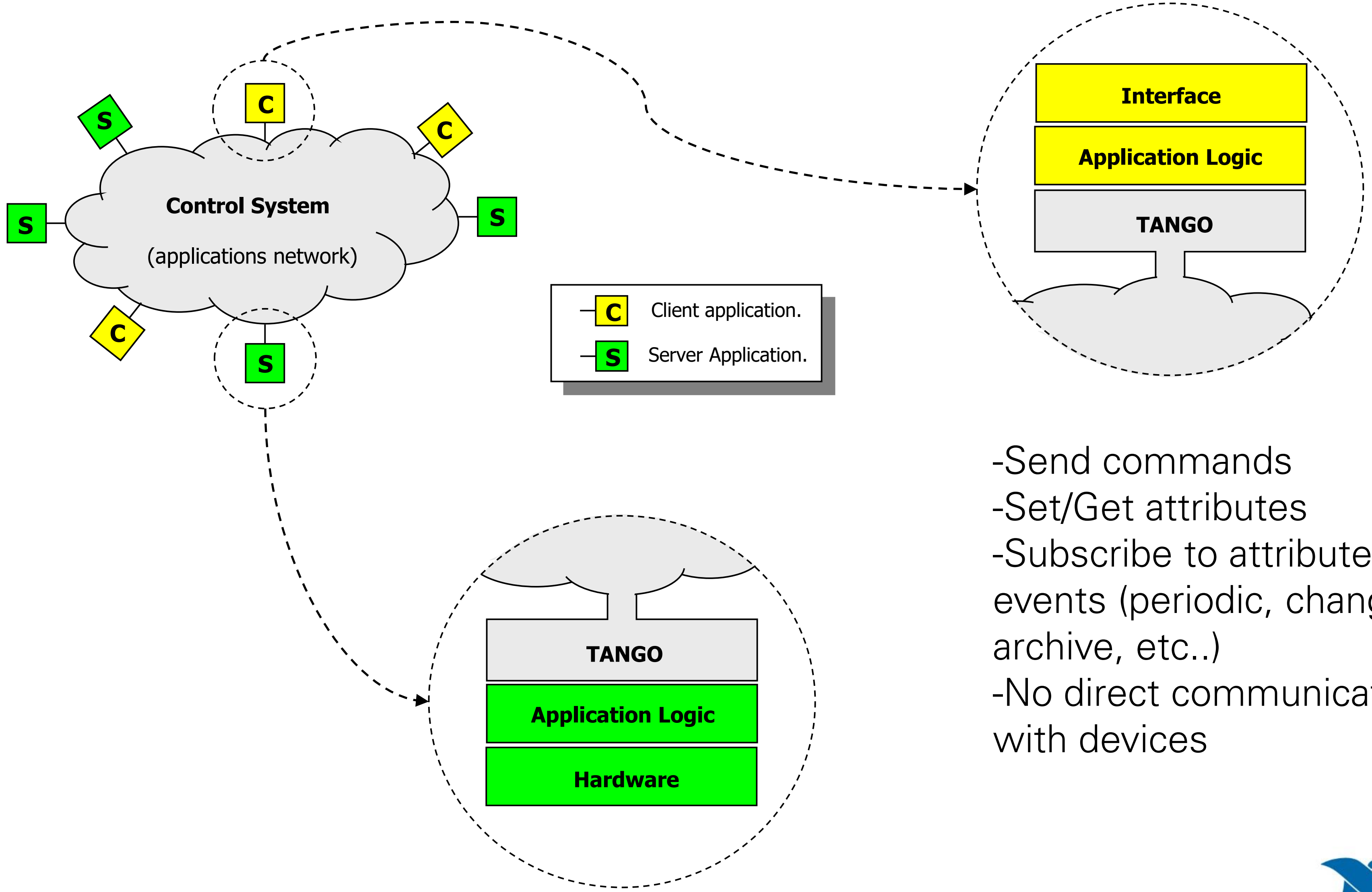
To be written by developers

Supplier driver

POGO:
- C++/Java/Python glue code
- Basic HTML doc

NATIONAL INSTRUMENTS™

Clients/Servers

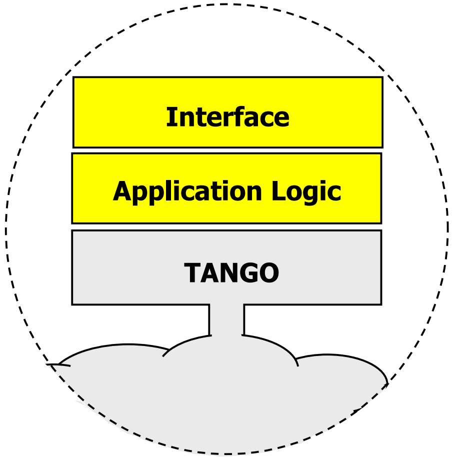


-Send commands
-Set/Get attributes
-Subscribe to attributes events (periodic, change, archive, etc..)
-No direct communication with devices

TANGO-NI interface

LabVIEW client (binding)

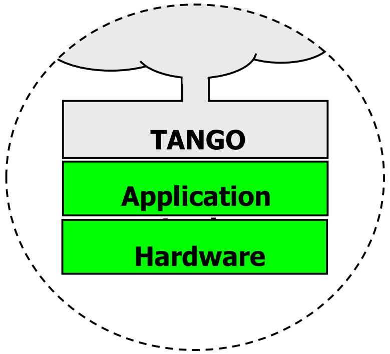
Name
- _Argin.vi
- _Argout.vi
- _AttributeInfo.vi
- _AvExtractAsCluster.vi
- _AvExtractAsValue.vi
- _AvInsertFromSimpleValue.vi
- _CommandInfo.vi
- _TangoAttributeConfigEventGroupCreate.v
- _TangoAttributeConfigEventGroupKill.vi
- _TangoAttributeConfigEventHandler.vi
- _TangoAttributeDataReadyEventGroupCrea
- _TangoAttributeDataReadyEventGroupKill.
- _TangoAttributeDataReadyEventHandler.vi
- _TangoAttributeEventGroupCreate.vi
- _TangoAttributeEventGroupKill.vi
- _TangoAttributeEventHandler.vi
- _TangoDeviceAttributesInfo.vi
- _TangoDeviceAttributesList.vi
- _TangoDeviceCommandsInfo.vi
- _TangoDeviceCommandsList.vi
- _TangoDeviceStateAndStatus.vi



- Developed by Synchrotron SOLEIL

TANGO-NI HW bridge (Device Server)

3 options:

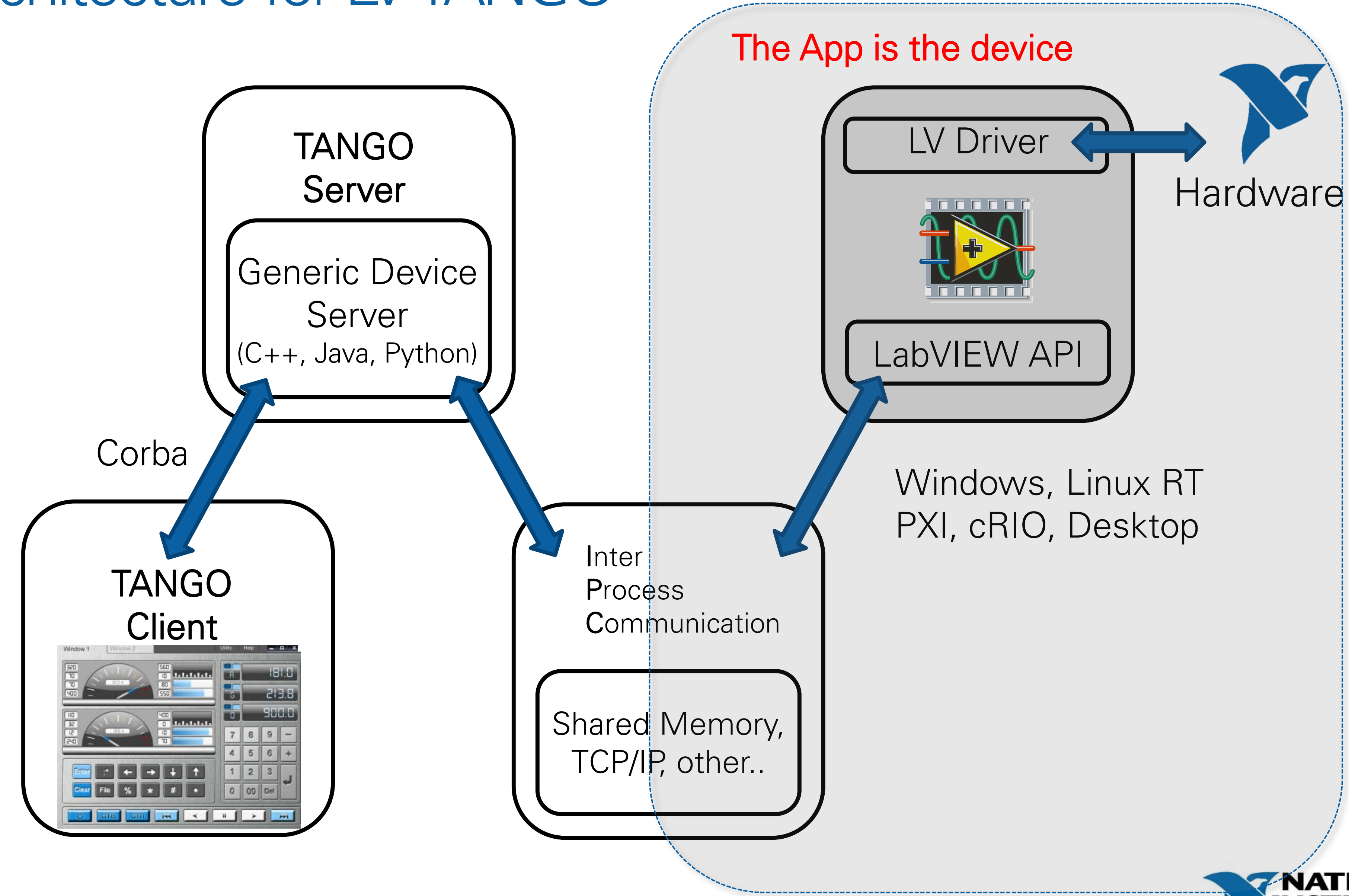


- TANGO Server and NI HW are on the same machine & drivers are available
  - One can develop his Device Server for NI HW by creating a class for this HW and calling its NI driver

- TANGO Server and NI HW are NOT on the same machine or no drivers available or preferred programming IDE
  - One can do the same and calls will be done remotely through a chosen communication protocol (TCP/IP, etc..)

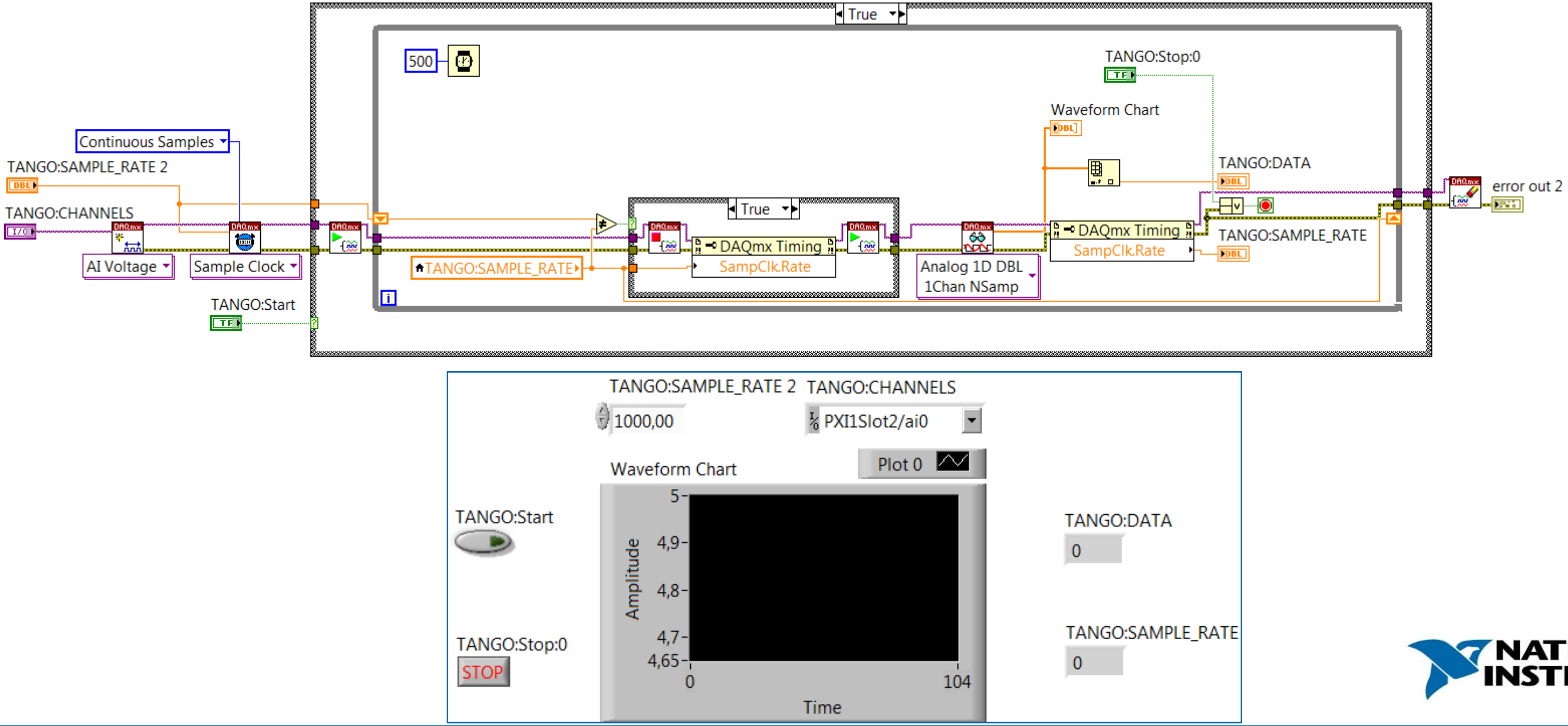
- TANGO-LV add-on (Developed by SOLEIL : http://www.tango-controls.org/downloads/bindings/)

Generic architecture for LV-TANGO

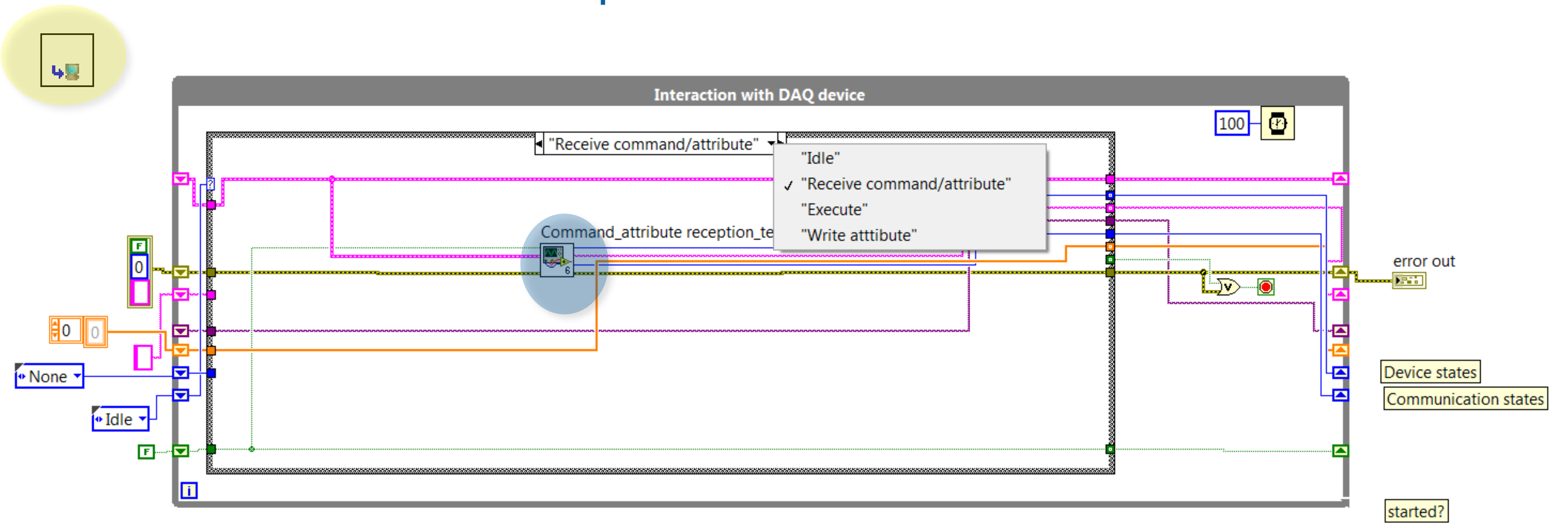


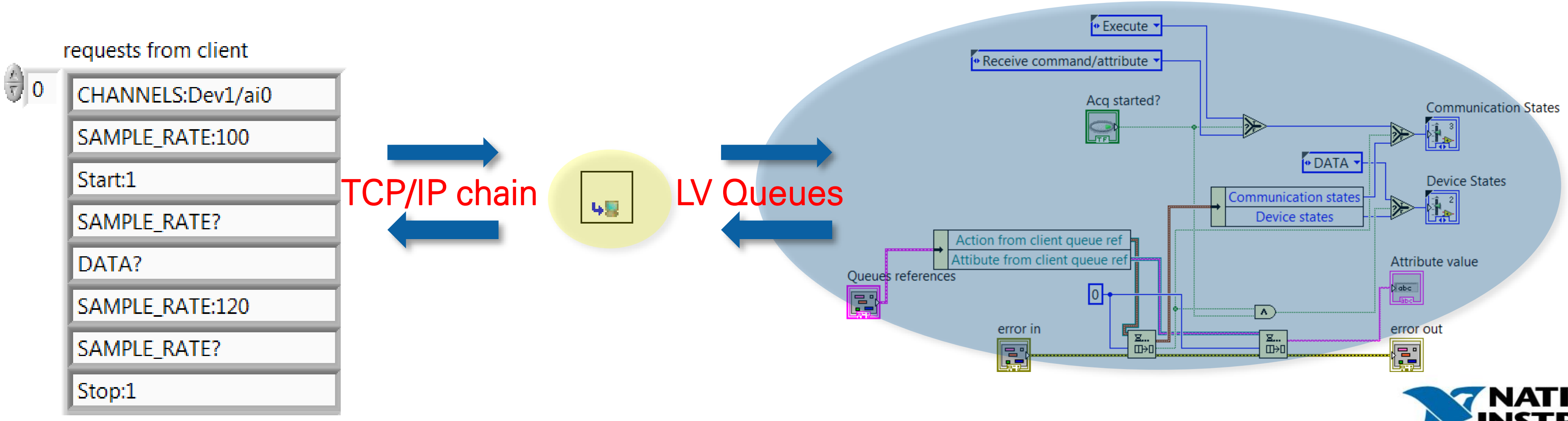
Remote Device Server : DAQ application example

- One has to adapt his LV application architecture (state machine, events,…) to:
  - communicate with the client (via the Device Server) by using a communication protocol instead of LV front panel controls
  - make some attributes available at any time if needed



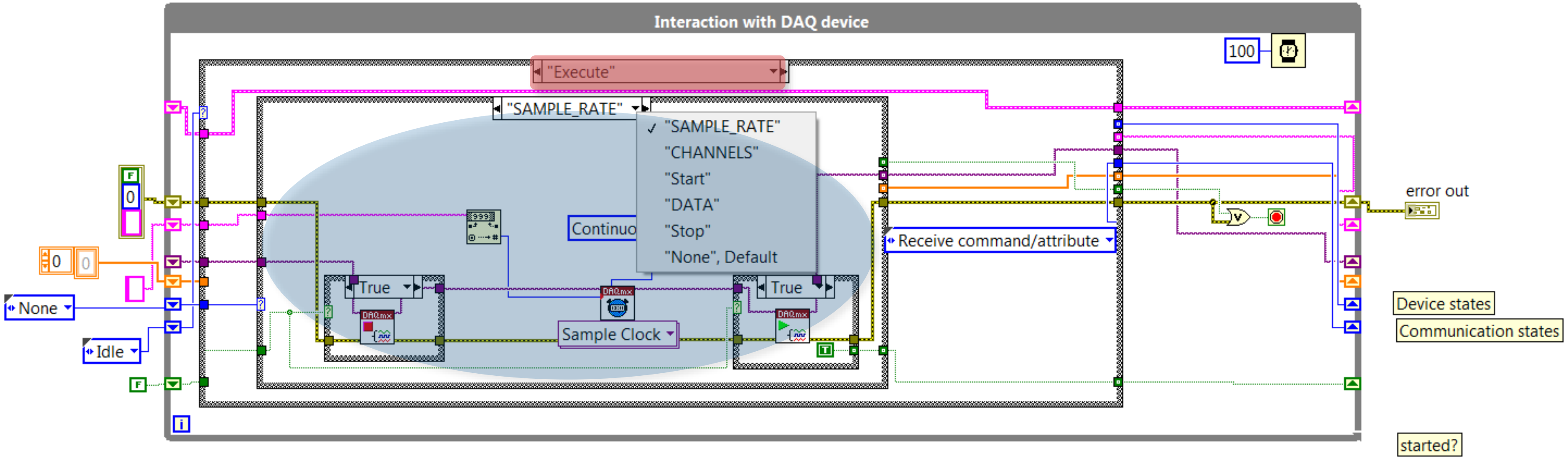
State machine: Get client requests

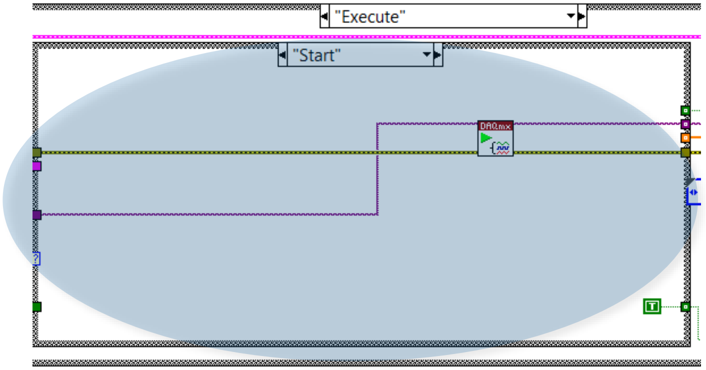


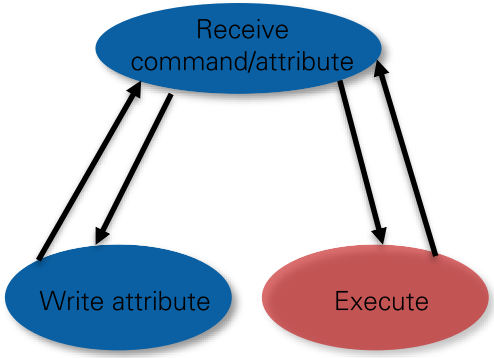


State machine: Handle client requests

Set attributes/Run actions



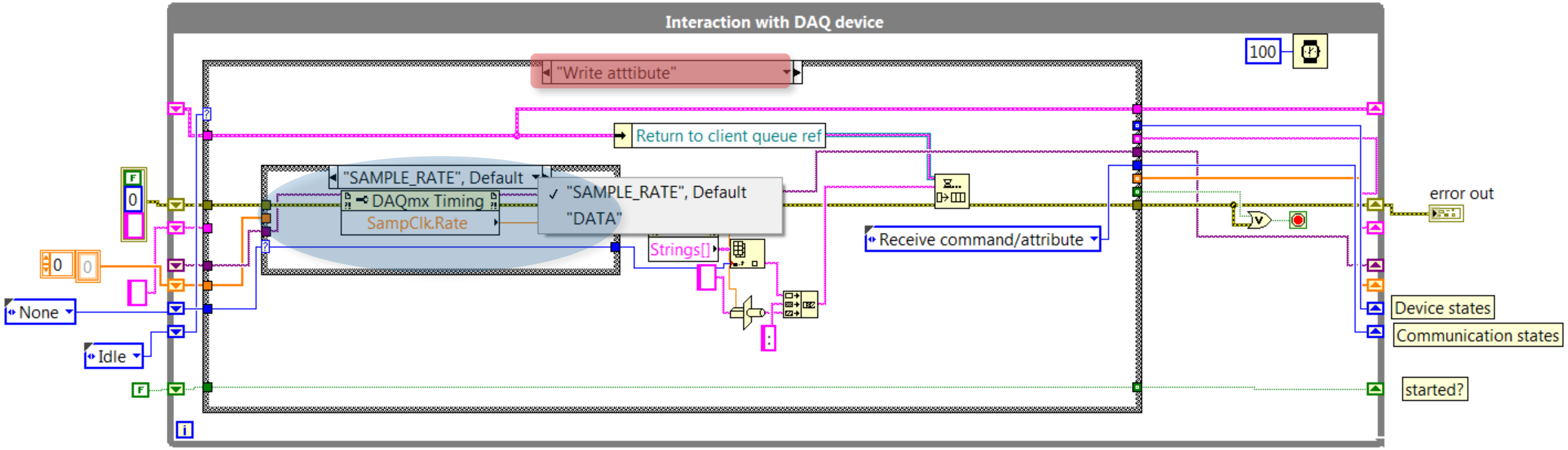


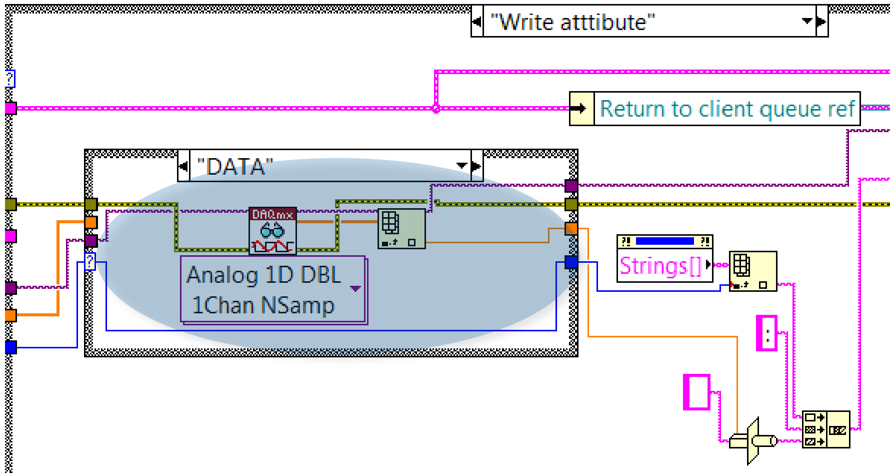


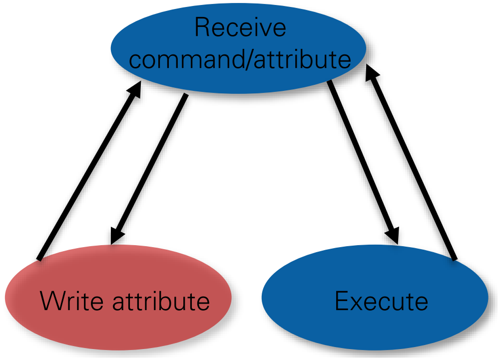
NATIONAL INSTRUMENTS

State machine: Handle client requests

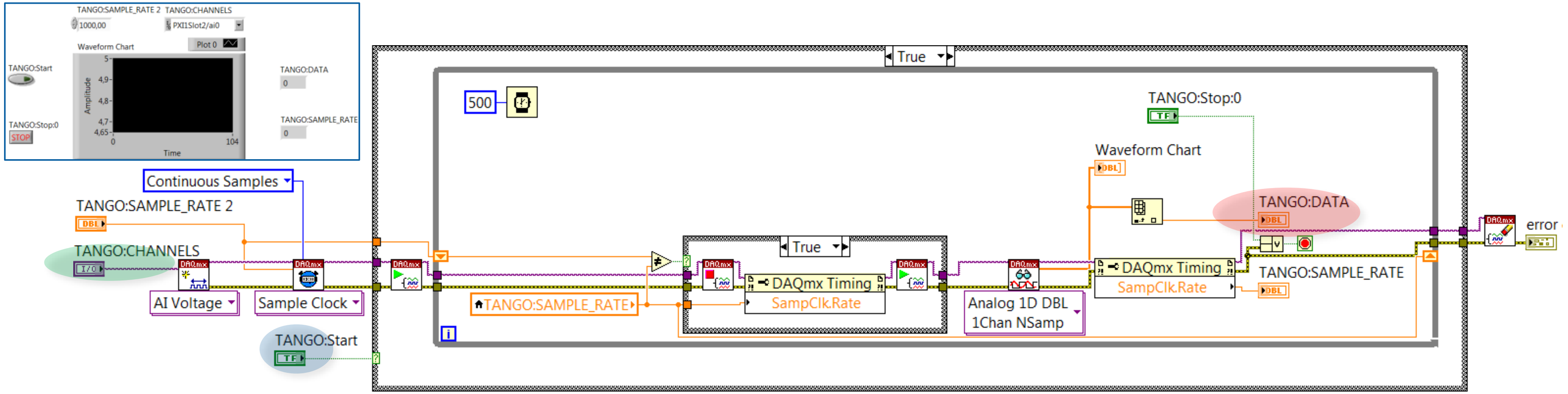
Get attributes values



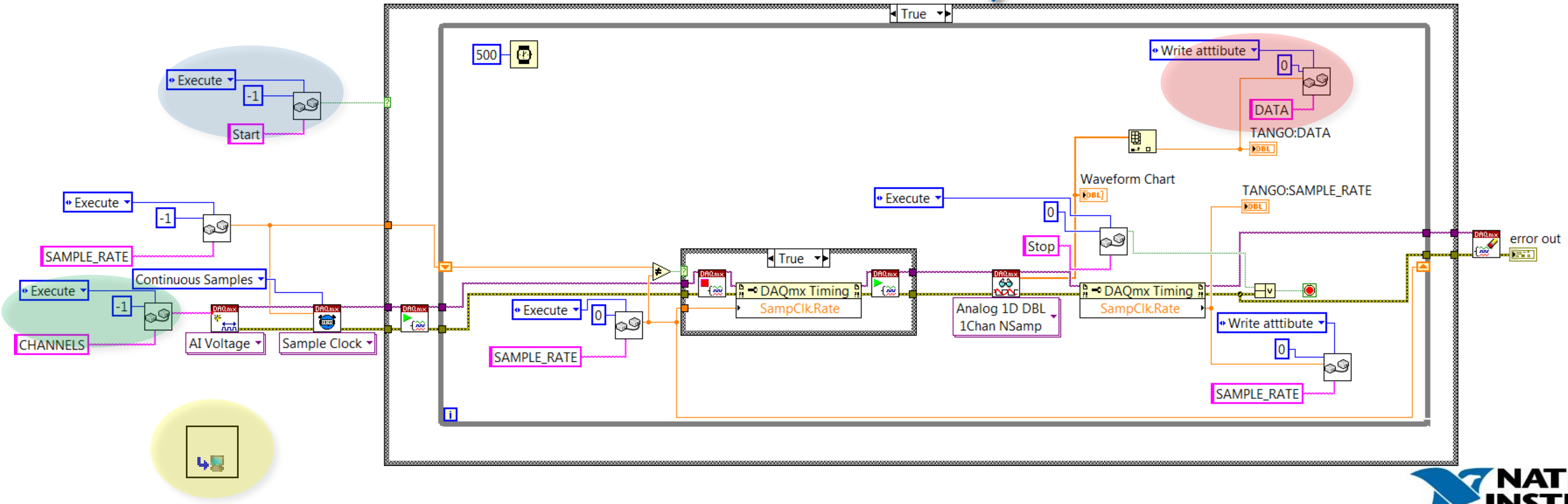




Automatic conversion: no data flow change



Automatic conversion : VI Scripting



NATIONAL INSTRUMENTS

State machine vs automatic conversion

Architecture	Pros	Cons
State machine	• Modular, scalable • Good graphical understanding	• Need to "rethink" the application and break into sub-steps
Automatic conversion	• Very quick method • Minor changes to the original application	• Not very modular (new actions, attributes) • All attributes/actions evaluated at each iteration • Reconfiguration not easy

NATIONAL INSTRUMENTS™

Conclusion

- TANGO is an object oriented middleware where devices belong to classes (Commands, attributes).
- Several tools exist to assist developers in creating their device classes
- Device servers can access directly or remotely to HW.
- The application accessing to HW has to be structured according to the foreseen commands/attributes.
- One can also think about a generic device server targeting LV applications with generic templates….

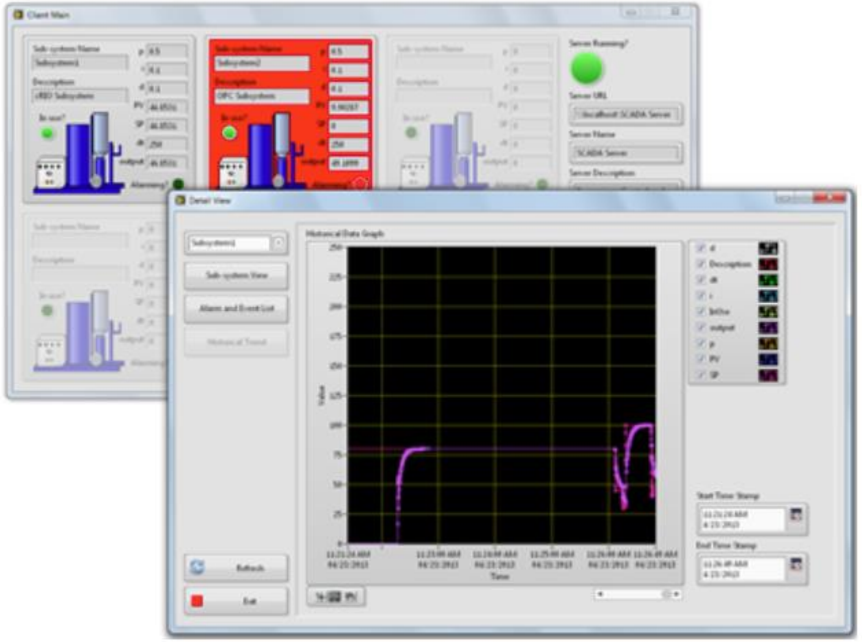
Use LV to communicate with 3rd party PLCs and extend the I/Os

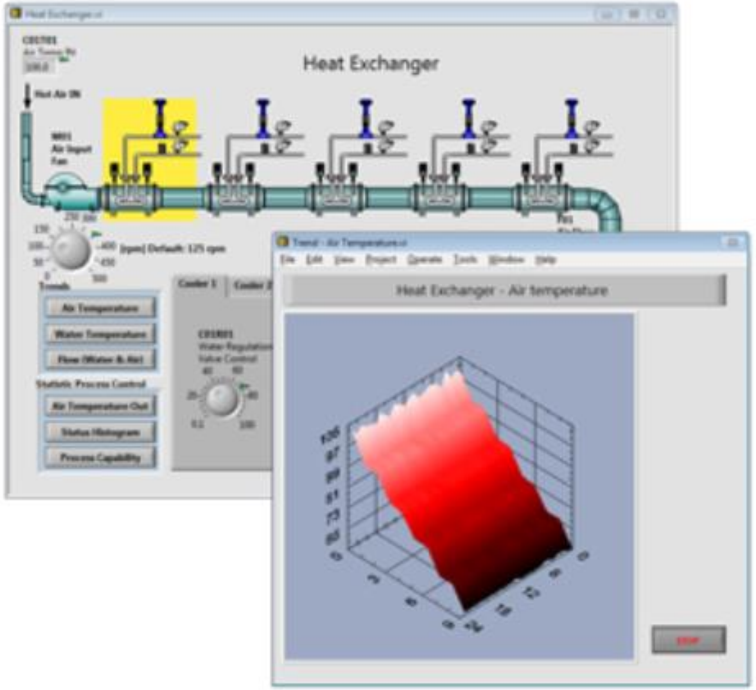
Agenda

- LV DSC
- Shared variables
- EPICS
- OPC
- Modbus
- Ethercat

SCADA systems : LV DSC

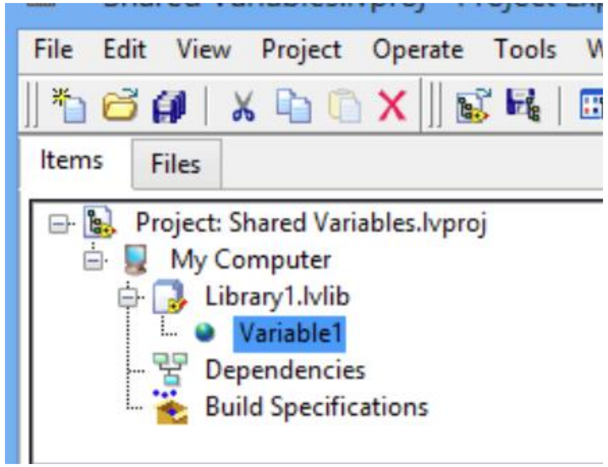
The LabVIEW Datalogging and Supervisory Control (DSC) Module is the ideal LabVIEW add-on for developing your **HMI/SCADA or high-channel-count data-logging** applications. With LabVIEW DSC, you can interactively develop a **distributed monitoring and control system** with tags ranging from a few dozen to tens of **thousands**. It includes tools for **logging data** to a networked historical database, tracking real-time and historical trends, **managing alarms and events**, networking LabVIEW Real-Time targets and OPC devices into one complete system, and adding security to user interfaces.

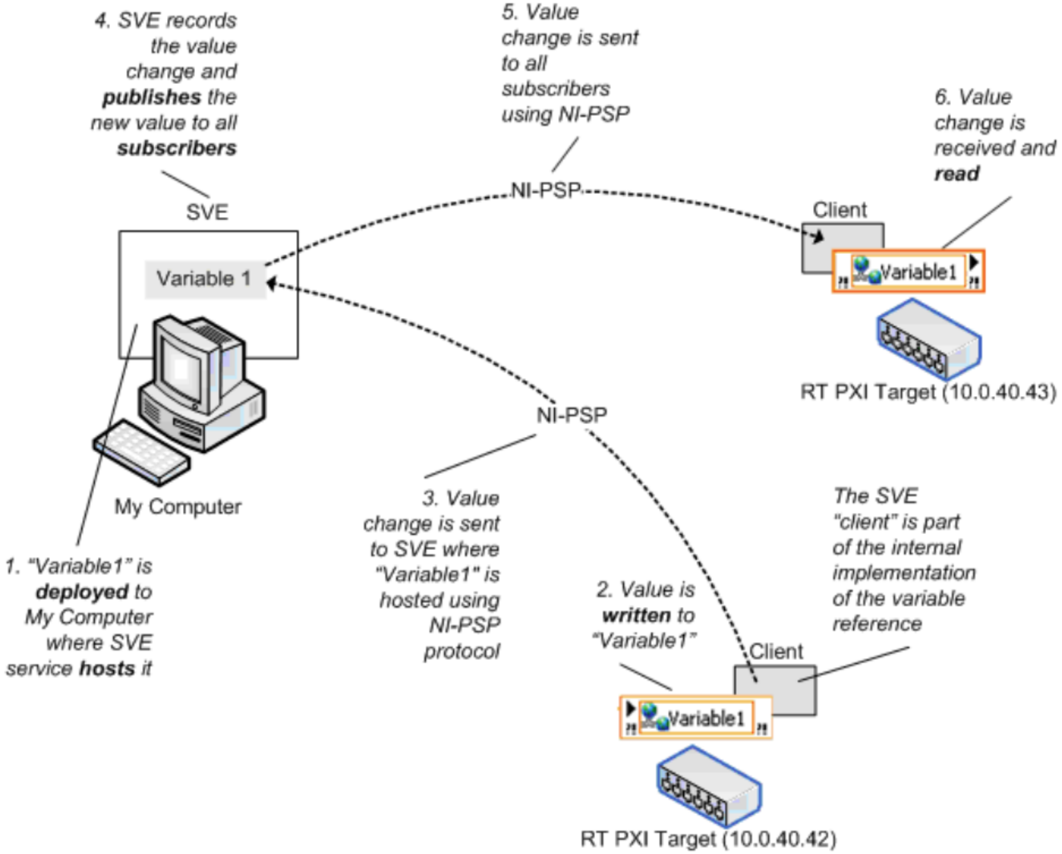




Shared variables

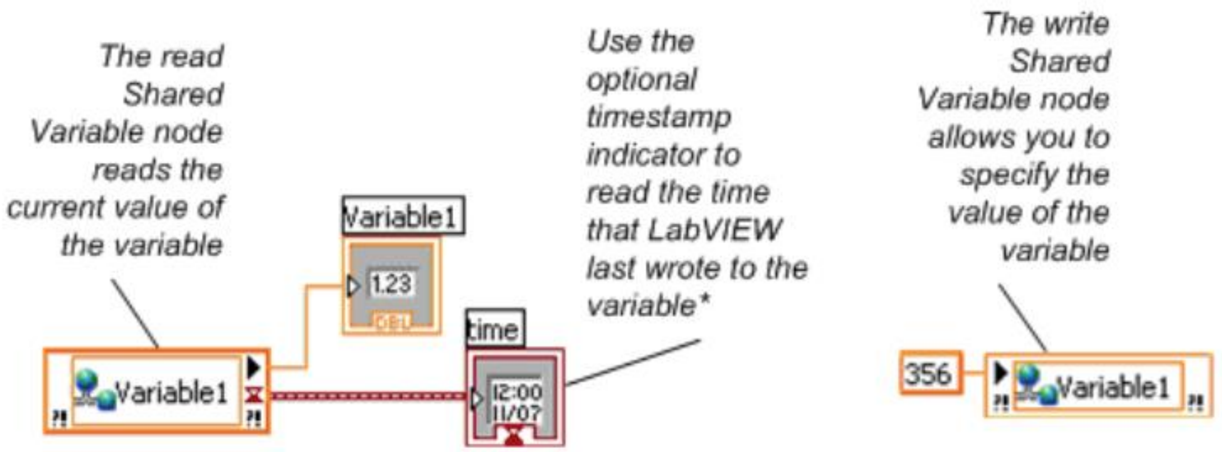
NI-PSP is National Instruments' proprietary publish-subscribe protocol (PSP). NI-PSP is composed of a server called the Shared Variable Engine that hosts values, timestamps, and other Shared Variable information. NI-PSP is designed for the use case where many accessors must access or update a latest data value. It is not designed for high data throughput or low-latency.

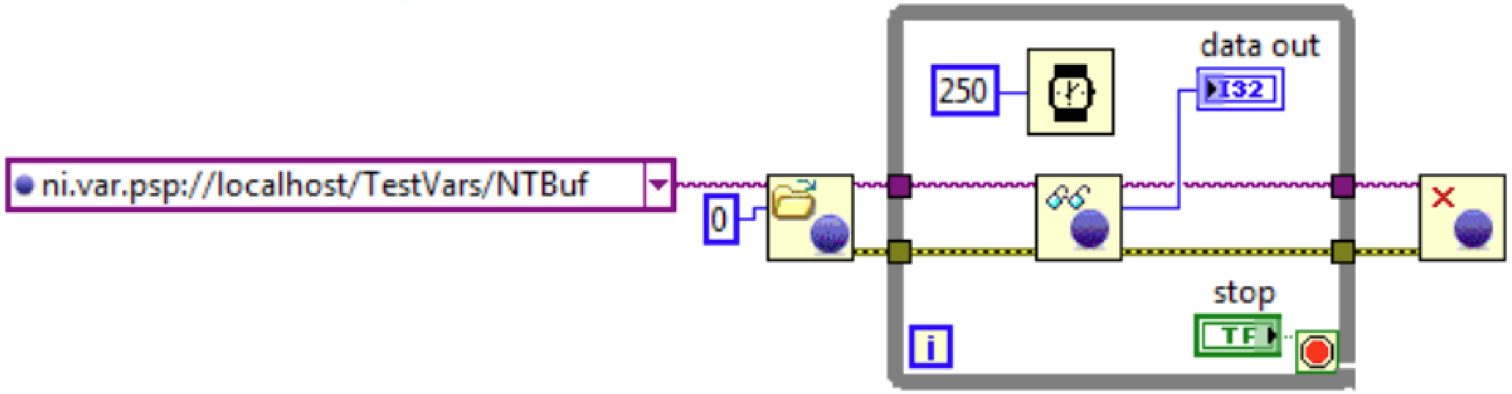


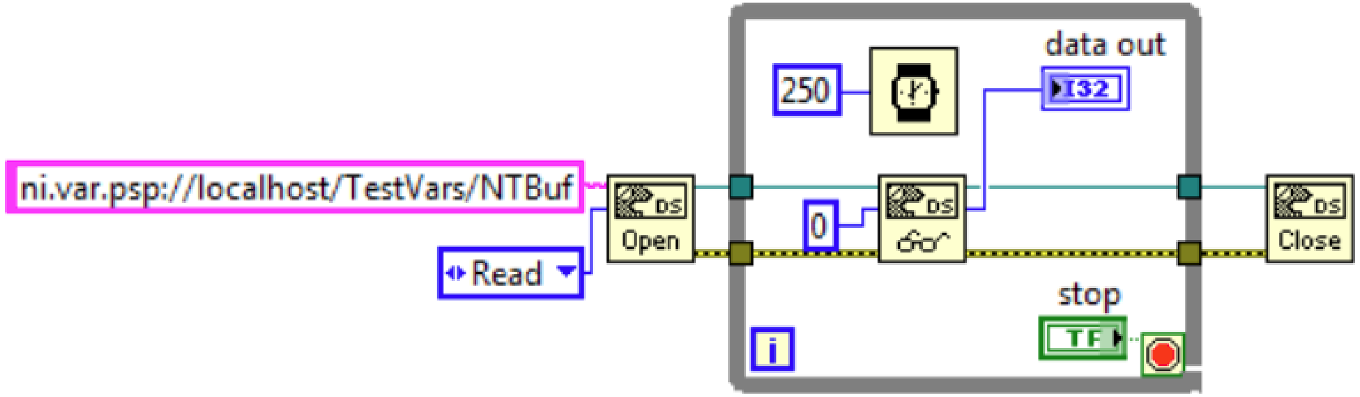


NATIONAL INSTRUMENTS™

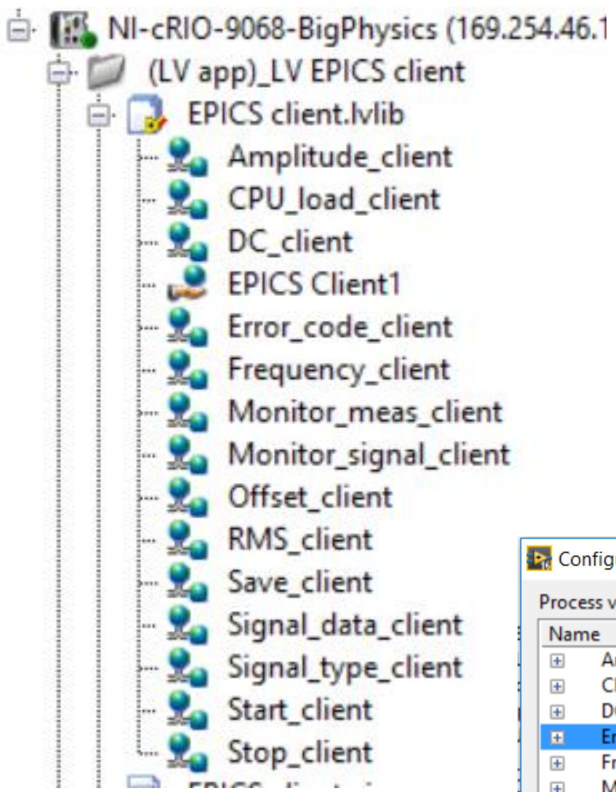
Shared variables

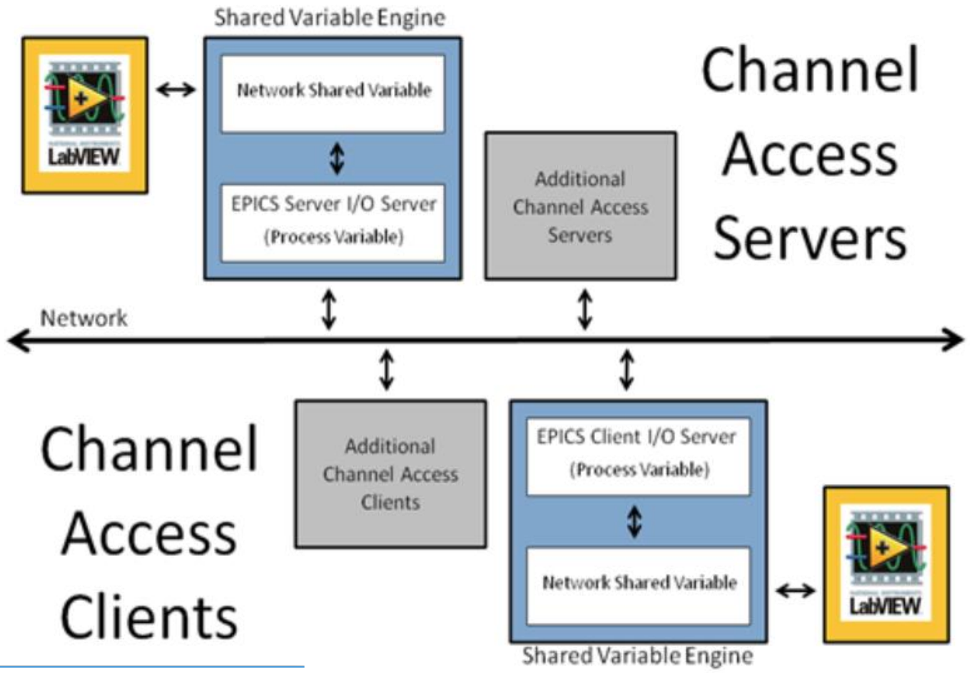


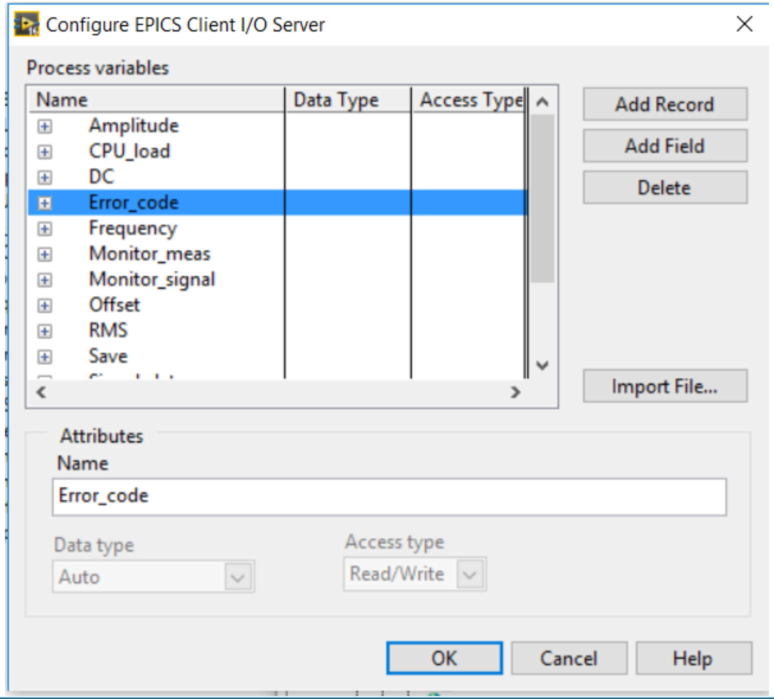




EPICS (DSC & RT)







NATIONAL INSTRUMENTS

OPC

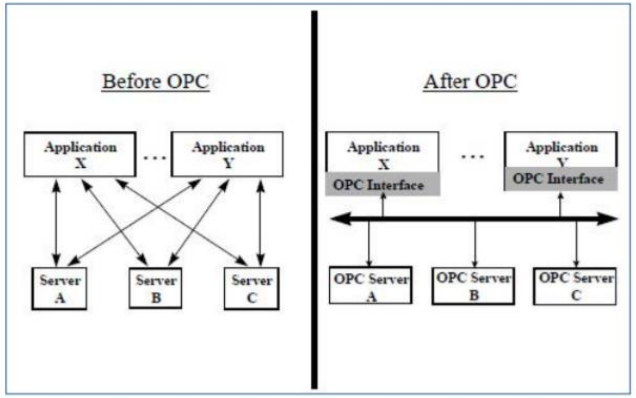
OPC, which is a Microsoft COM-based standard, allows client and server applications to communicate with each other. OPC is designed to be an **abstraction layer** between industrial networks and proprietary PLC drivers, and **multi-vendor interoperability**.

OPC is highly scalable and suited for high-channel-count systems.

- a tag gives a unique identifier to an I/O point (programmatically or user defined)

- Client software also specifies the rate at which the server supplies new data to the client. The client software does not need to perform time-consuming data polling (event-driven reactive object that waits for new data to arrive).

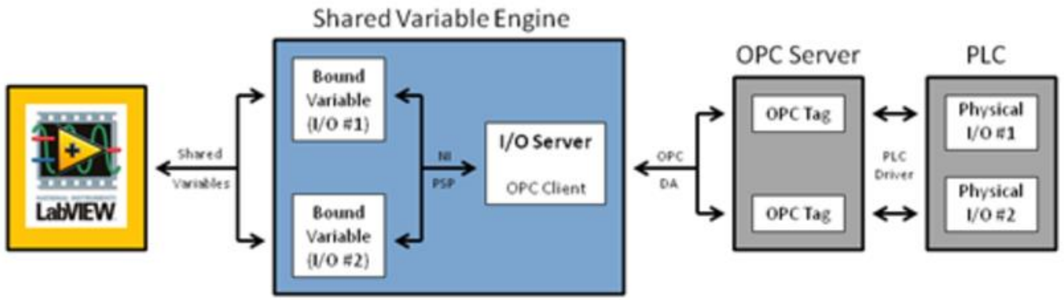
- The OPC server also provides alarm and event handling to client (operator parameters change, access violations, conditions etc…)



LV OPC Server I/O (LV DSC)

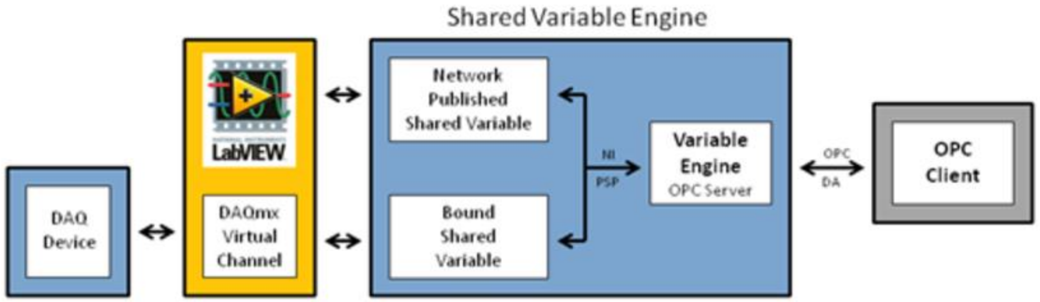
As a Client :

PLCs publish data to the network. An OPC Server program uses the PLC's proprietary driver to create OPC tags for each physical I/O on the PLC. NI OPC Servers contains a list of drivers for many of the industry's PLCs.



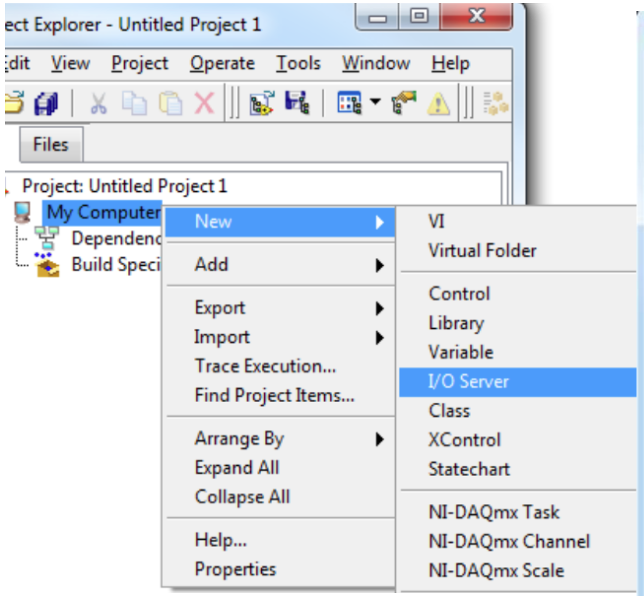
As a Server:

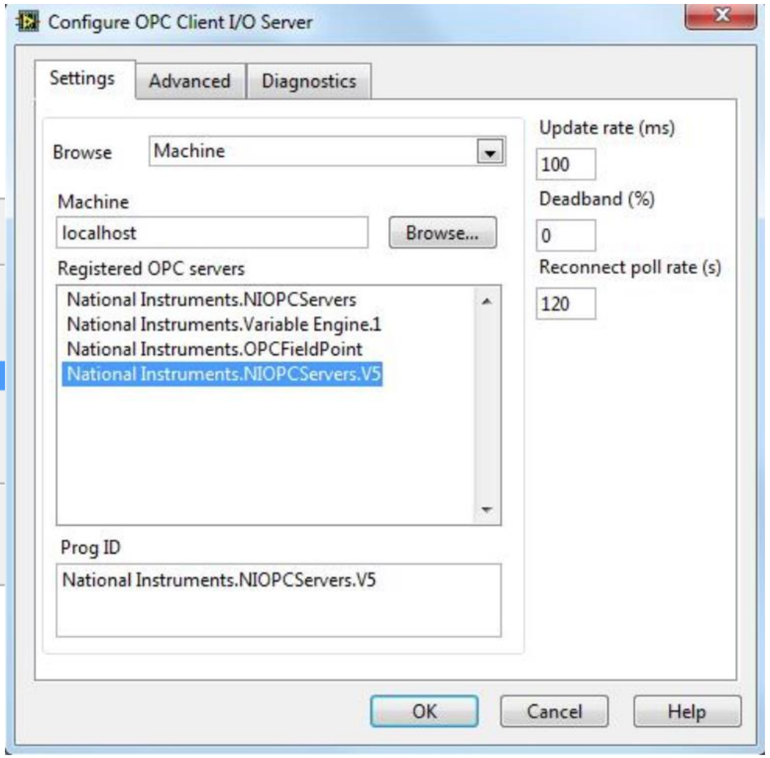
The SVE as an OPC server should not be confused with NI OPC Servers, because It does not contain proprietary PLC drivers. The SVE can take a network-published Shared Variable and create OPC tags that an OPC DA client can connect to.

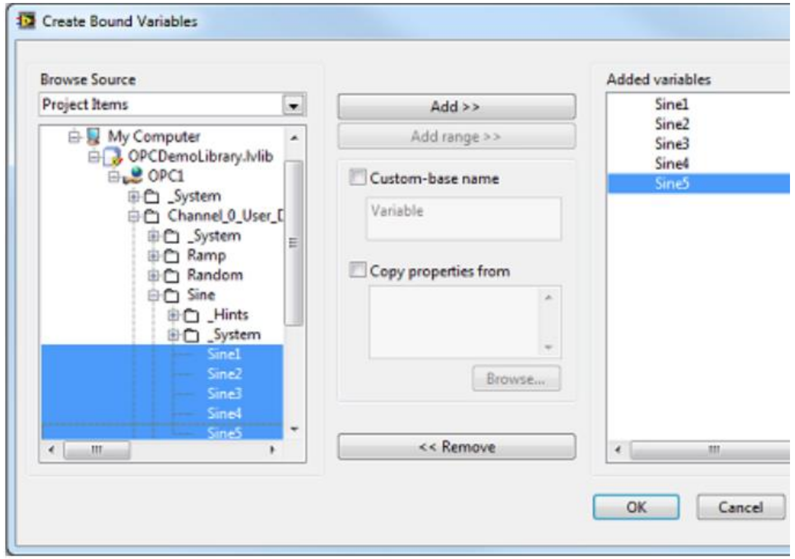


LV OPC Server I/O

LabVIEW allows developers to integrate with OPC systems. You can connect both OPC **clients and servers** to LabVIEW applications to share data. The primary component that allows LabVIEW to perform this action is the **Shared Variable Engine** (SVE).





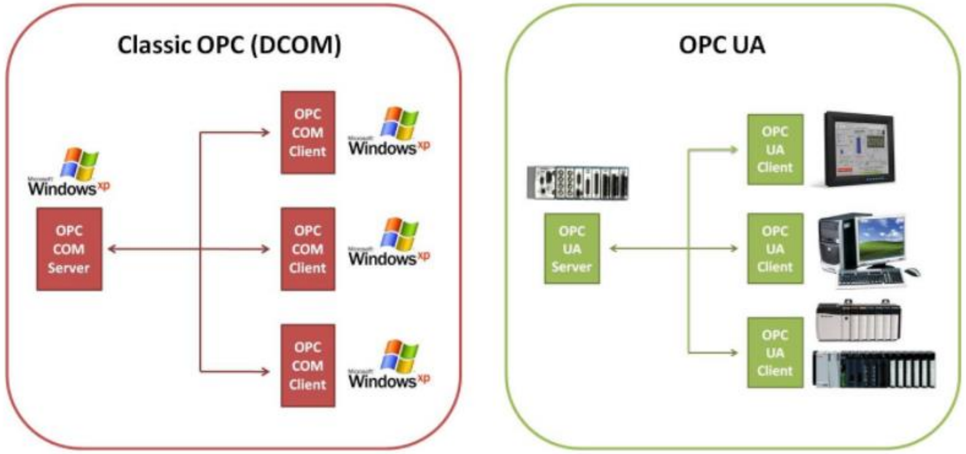


OPC UA

LabVIEW OPC UA Toolkit (or DSC or LV RT)

OPC Unified Architecture (UA) is a new communication technology standard. OPC UA includes all the functionality found in OPC Classic.
OPC UA is based on a **cross-platform**, business-optimized Service-Oriented Architecture (SOA), which expands on the security and functionality found in OPC.

- Expanded security (authentication and encryption)
- Easier IT integration (through firewalls, VPNs, etc..)
- Platform independance (Windows, OSX, Android, Linux etc.)
- Extensible (add new features with backward compatibility)



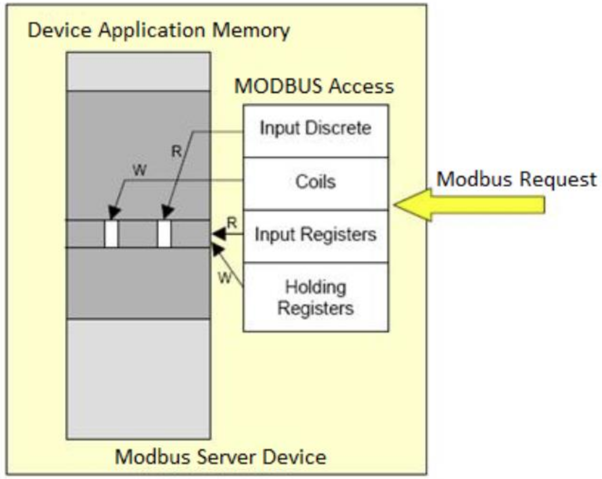
Modbus (DSC)

Modbus is a serial communication protocol published by Modicon in 1979 to communicate with PLC, and was then extended to the TCP protocol.

- The Modbus protocol follows a master/slave architecture. The master transmits a request to a slave and waits for the response. The frame has Slave ID (or IP), R/W, data, and CRC (for serial).
- Modbus supports two data types: a Boolean value and an unsigned, 16-bit integer.
 For larger data types, slaves split data into registers (for example, a pressure sensor split a 32-bit floating point value across two 16-bit registers).

Memory Block	Data Type	Master Access	Slave Access
Coils	Boolean	Read/Write	Read/Write
Discrete Inputs	Boolean	Read-only	Read/Write
Holding Registers	Unsigned Word	Read/Write	Read/Write
Input Registers	Unsigned Word	Read-only	Read/Write

Table 1. Modbus Data Model Blocks

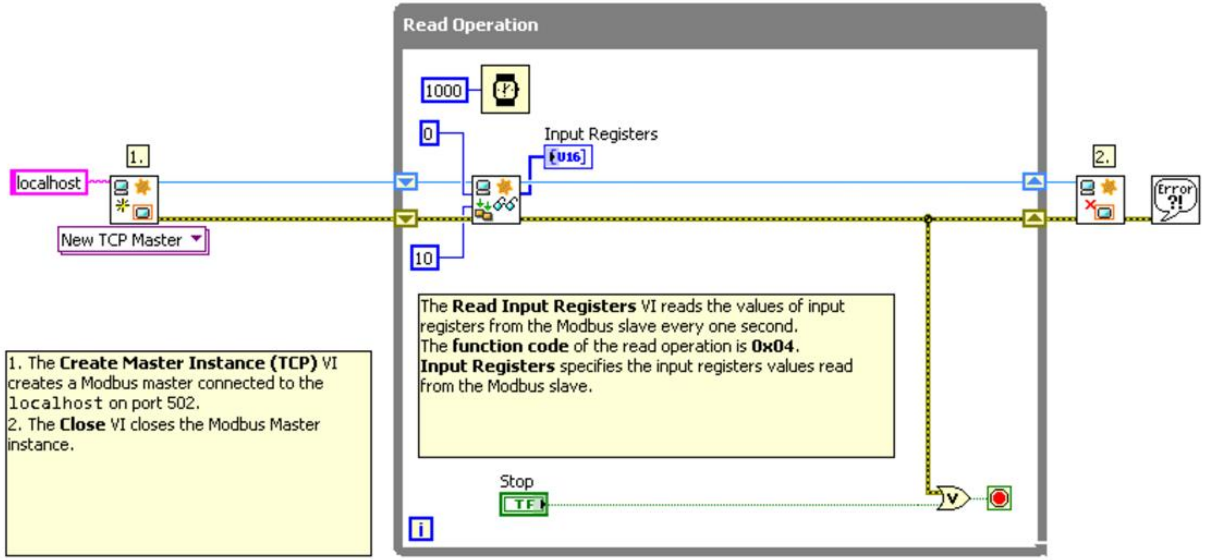


Modbus interfaces

NI provides three primary mechanisms for interfacing with Modbus devices: (1) a high-level OPC server, (2) a Modbus I/O server, and (3) a low-level Modbus API .

Low level API

Preferred option when your application needs flexibility or a high level of control over the sequencing and timing of Modbus requests.
The flexibility and power offered by this API also means your application code must be more complex to correctly manage the API



[+] Enlarge Image

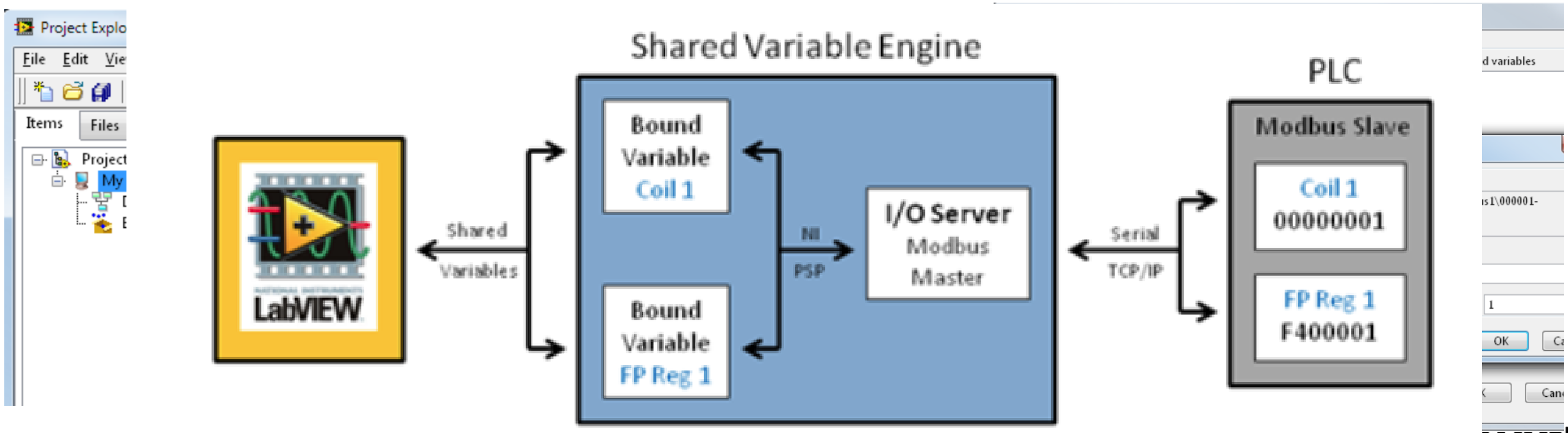
Figure 3. Master on RT Target.vi

Modbus I/O Servers

Modbus I/O servers (LabVIEW DSC and LabVIEW Real-Time), provide a high-level engine.

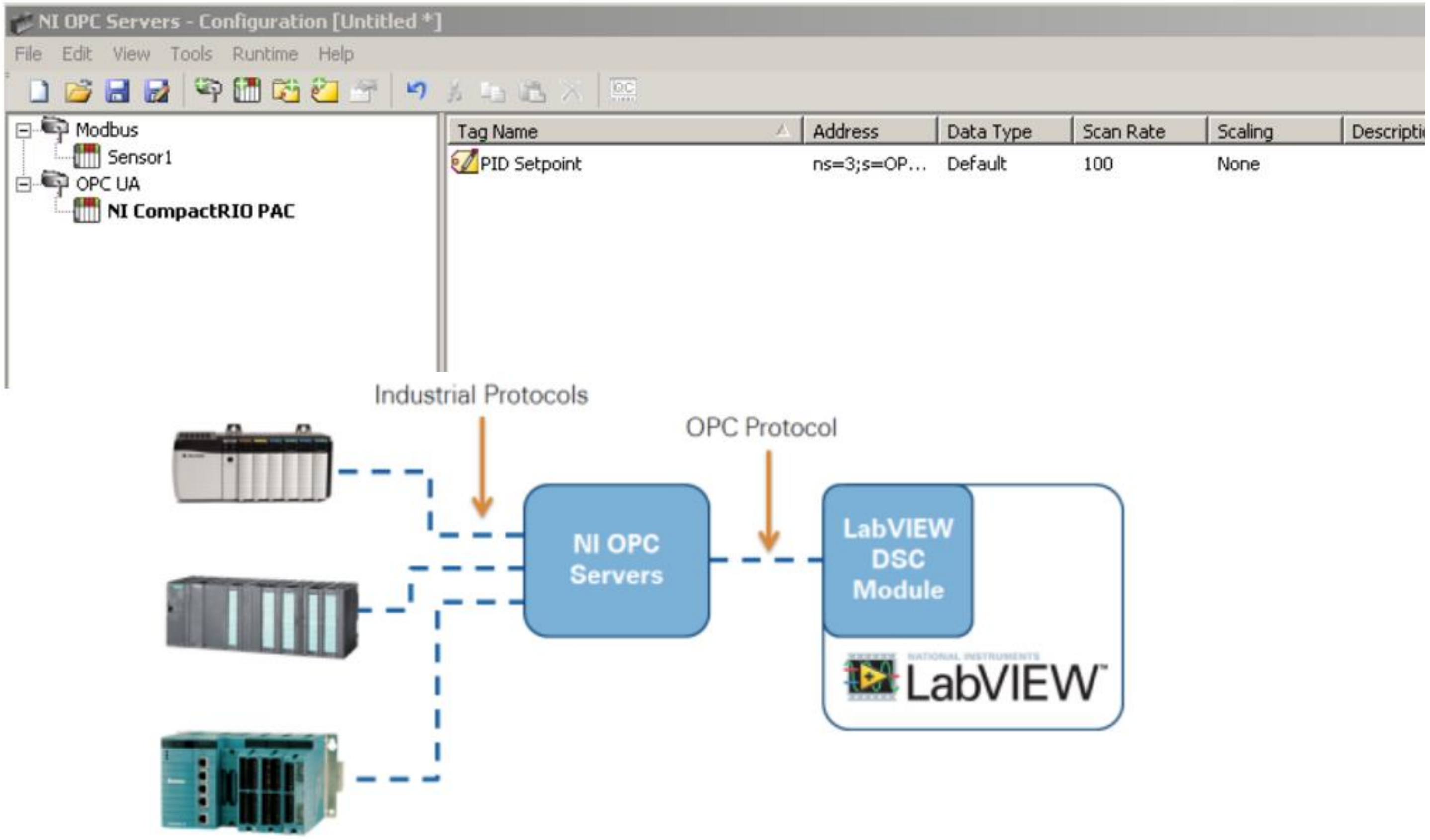
You register the set of data you would like to access and the I/O server schedules the requests automatically at the specified rate, without specifying a function code.

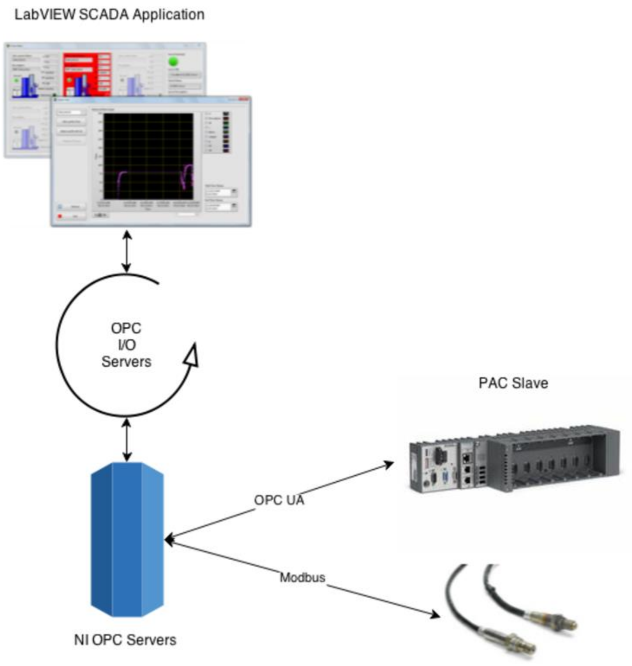
After the I/O server is created, you may specify the items on the device you wish to read. For example, you can read the holding register at address 0 by mapping a variable to the item 400001, read the first bit of this register by selecting 400001.1, and read the single precision float that is stored in registers 0 and 1 by selecting F400001.



NI OPC Servers for Modbus

For complicated applications involving many slave devices that communicate over different protocols, the standard Modbus I/O might not suffice. A common solution is to use an OPC server, which acts as a **data aggregator** for all of your systems, and then use the OPC I/O servers included in the LabVIEW DSC Module to communicate with that OPC server.

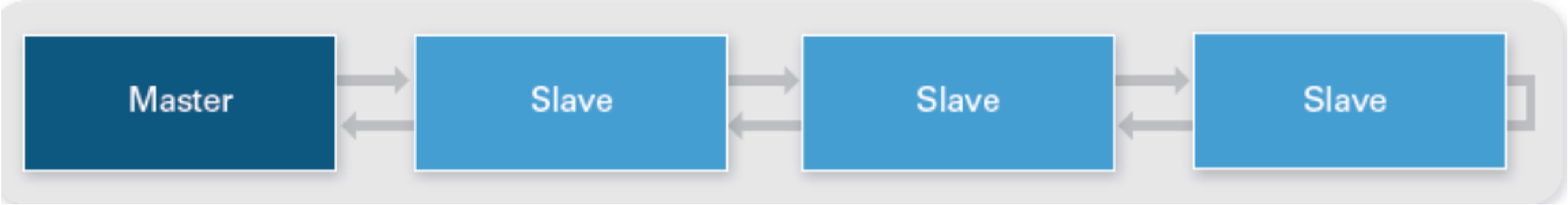




Ethercat

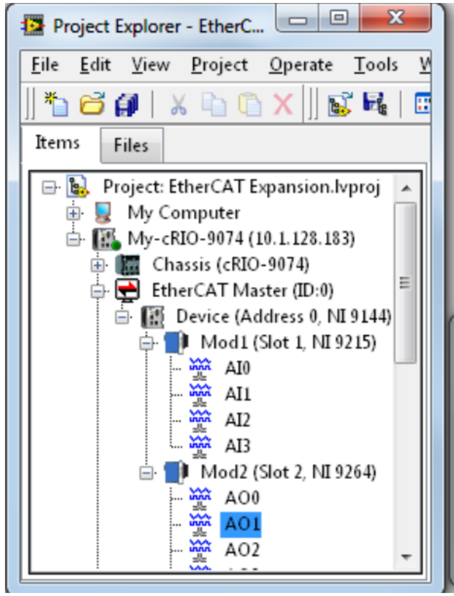
EtherCAT is a high-performance, **real-time Ethernet** protocol that uses a master/slave architecture that daisy chains into a line topology over standard Ethernet cabling.
EtherCAT is designed to achieve **high-Speed performance and high channel counts for single-point applications** such as control.
Another factor in achieving deterministic networks is the master controller's responsibility to synchronize all slave devices with the same time using **distributed clocks**.

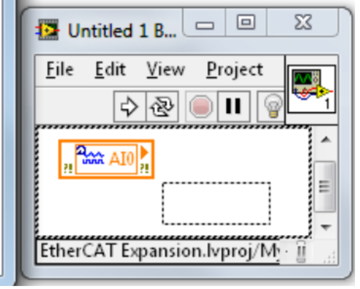
Data is communicated between master and slaves in the form of process data objects (PDOs). Each PDO has an address to one particular slave or multiple slaves, and this "data and address" combination makes up an EtherCAT telegram.
These Ethernet frames are processed **on the fly**.
When device 1 encounters the Ethernet packet sent by the master, it automatically begins streaming the packet to device 2, all while reading and writing to the packet with only a few **nanoseconds of delay.**

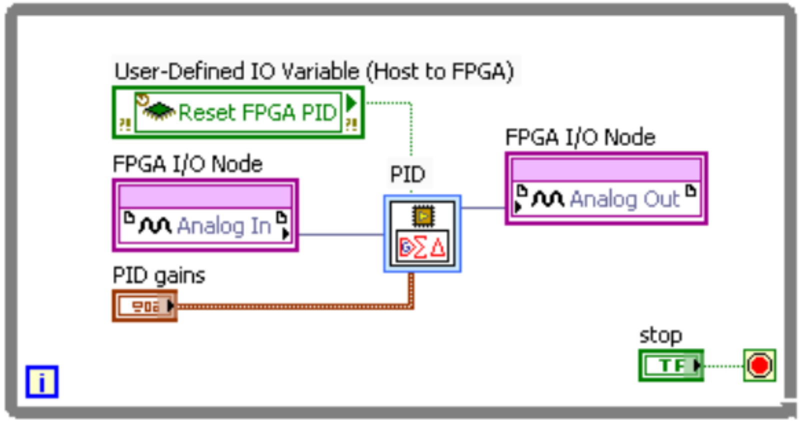


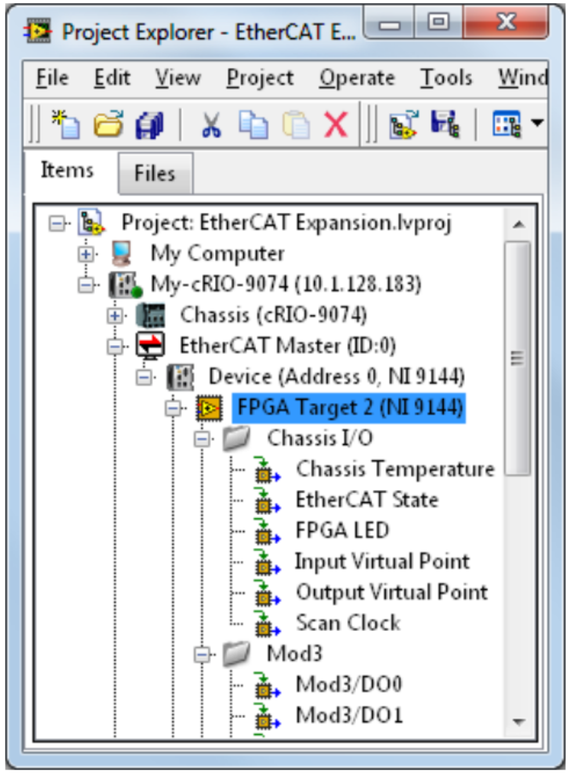
NI Ethercat

NI has Ethercat masters and slaves.
An NI Slave can be controlled by a 3rd party master, and vice versa.









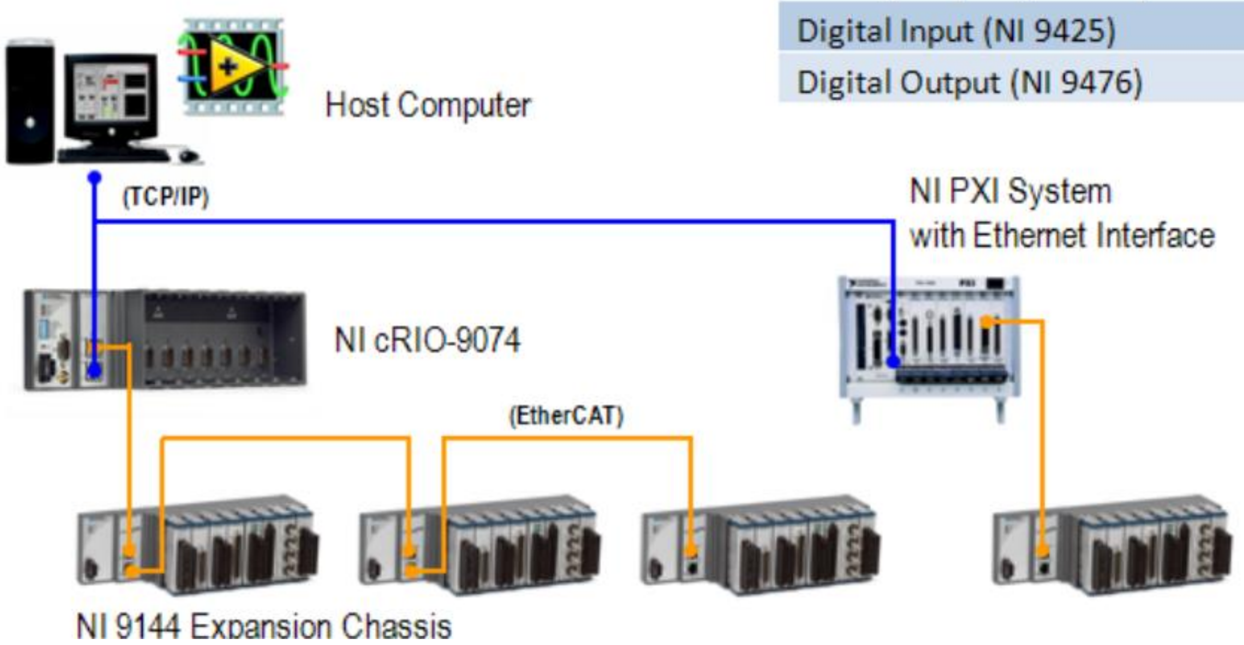
NATIONAL INSTRUMENTS™

Ethercat performance

Synchronization	<1µs
Throughput	12,5MB/s
Distance	100m before repeater

cRIO-9022	256 channels (1 full chassis)	1024 channels (4 full chassis)
Analog Input (NI 9205)	1.43 ms	5.37 ms
Analog Output (NI 9264)	1.74 ms	6.62 ms
Digital Input (NI 9425)	1.28 ms	4.77 ms
Digital Output (NI 9476)	1.41 ms	5.32 ms

PXI-8106	256 channels (1 full chassis)	1024 channels (4 full chassis)
Analog Input (NI 9205)	0.19 ms	0.67 ms
Analog Output (NI 9264)	0.23 ms	0.80 ms
Digital Input (NI 9425)	0.14 ms	0.47 ms
Digital Output (NI 9476)	0.16 ms	0.53 ms



Benches examples

New products and trends for Big Physics

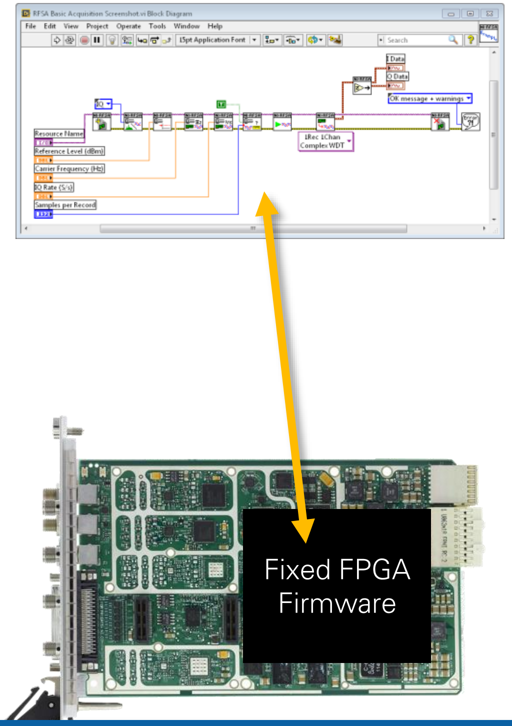
Agenda

- Software designed-instruments
- Deterministic ethernet & Synchronization
- Distributed system management (SystemLink)
- Linux support
- PXI Co-processing (PXImc)

Software-Designed Instruments

Typical Modular Instrument

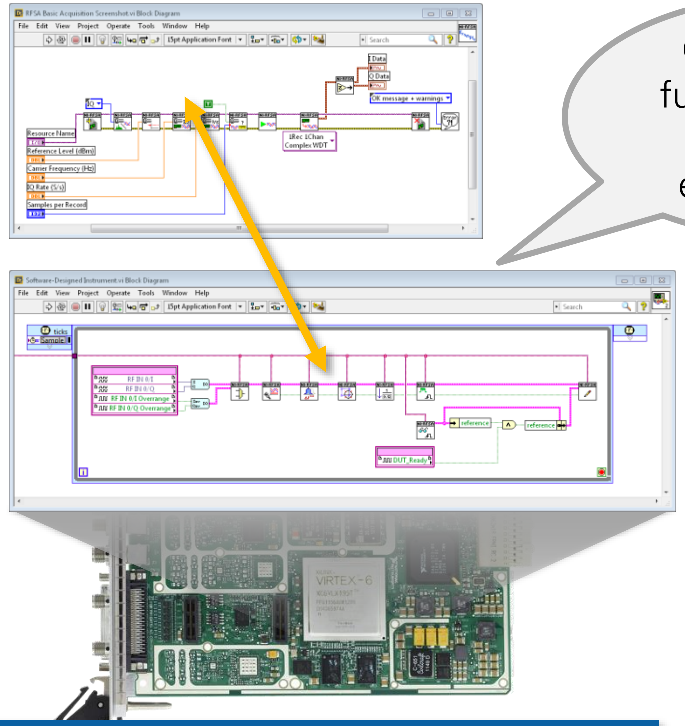
Software on the PC using the Instruments API (i.e. IVI or NI-Scope)



Software-Designed Instrument

Host-Software controlling Instrument through API or custom interface

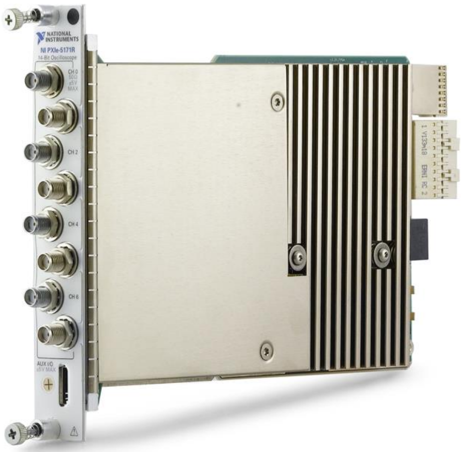
Out-of-the-box functionality with FPGA enhancements



FPGA-Code

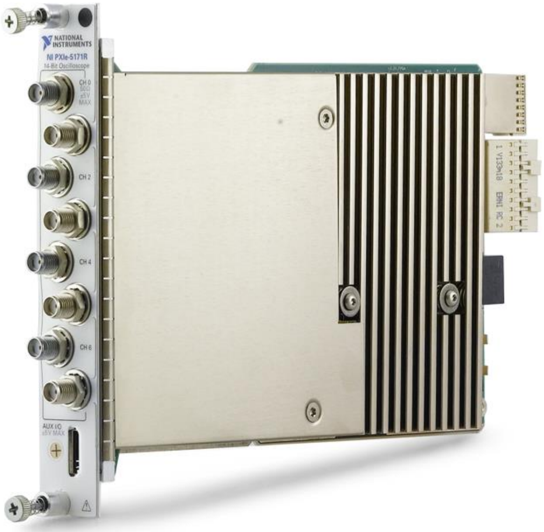
Similar hardware architecture and measurement quality

«open » instruments



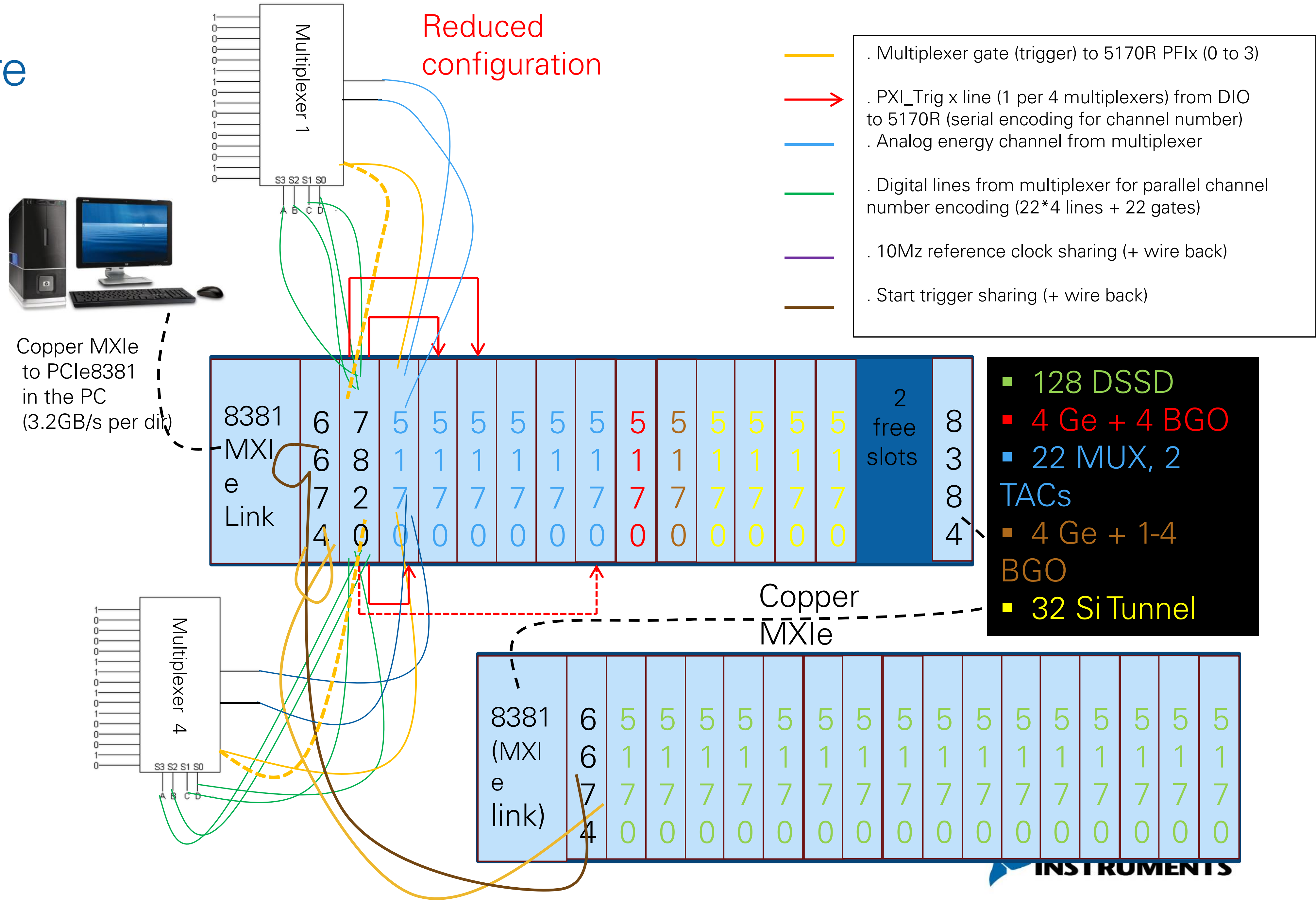
PXIe-517xR Variants

Specification	NI 5170R		NI 5171R
Channel Count	4	8	8
Full Bandwidth	100MHz		250MHz
Selectable Filters	-		100MHz
Open FPGA	K7 325T		K7 410T
Memory	750MByte	1.5GByte	1.5GByte
Resolution and Sample Rate	14-bit, 250MS/s		
Input Ranges	200mV_{pp}, 400mV_{pp}, 1V_{pp}, 2V_{pp}, 5V_{pp}		
Input configuration	50Ω, selectable AC or DC coupling per channel		
Analog performance (preliminary data)	~ -78dBc (30MHz signal, anti-alias filter enabled) >10 ENOB (full bandwidth, 0.4V_{pp} – 5V_{pp} ranges) 11 ENOB (anti-alias filter enabled, 0.4V_{pp} – 5V_{pp} ranges)		
Part-Number Price (USD) Price (EUR)	783690-01 $6,999 €6.350	783691-01 $9,999 €9.070	783692-01 $11,999 €10.900

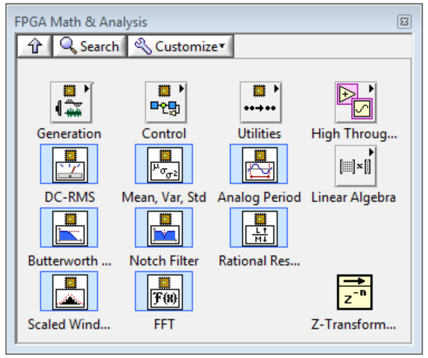


IN2P3 architecture

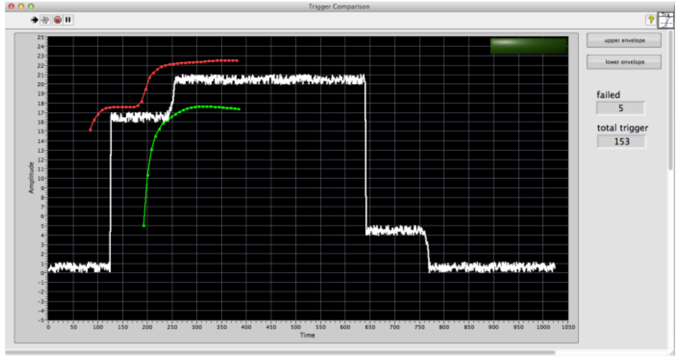
224 channels



Use cases



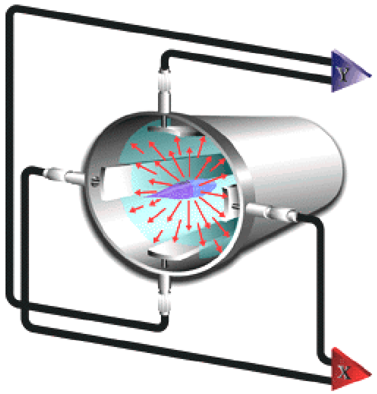
- Workaround to bandwidth or CPU performance limitation
- Create advanced triggers (frequency content, channels combination etc…) and avoid unuseful data
- Implement custom on-the-fly processing (averaging, filtering, timestamping etc..)
- Lower testing times (control loops with P2P,
 DUT control with digitial lines….)
- Continues processing without dead-time missing events



Application Example: Beam Position Monitoring

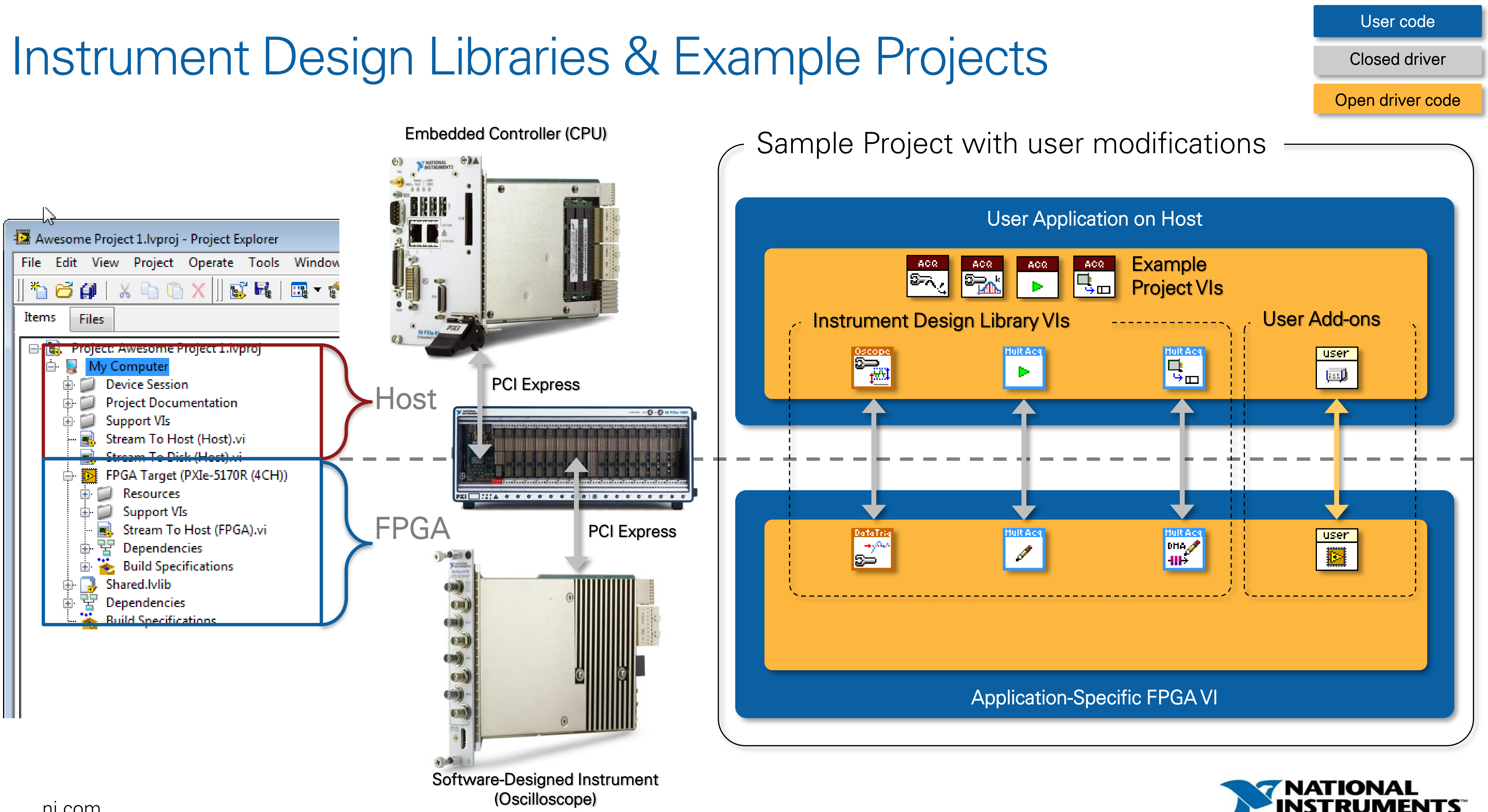
- Beam Position Monitor (BPM) measure the position of a particle beams in closed loop accelerators or storage machines while a BPPM in addition also measures the phase-relationship of particles to a RF-wave

- Acquire the signals of 250~500 MHz on 4 channels (±X, ±Y)
- Down-convert the RF-signals
- Extract position and phase-information from signal
- Provide position-data to control system



- <u>Benefits of the new NI PXIe-5171R</u>
  - High Channel density (8 channels per PXI-slot)
  - Signal processing in FPGA for time & frequency data in parallel

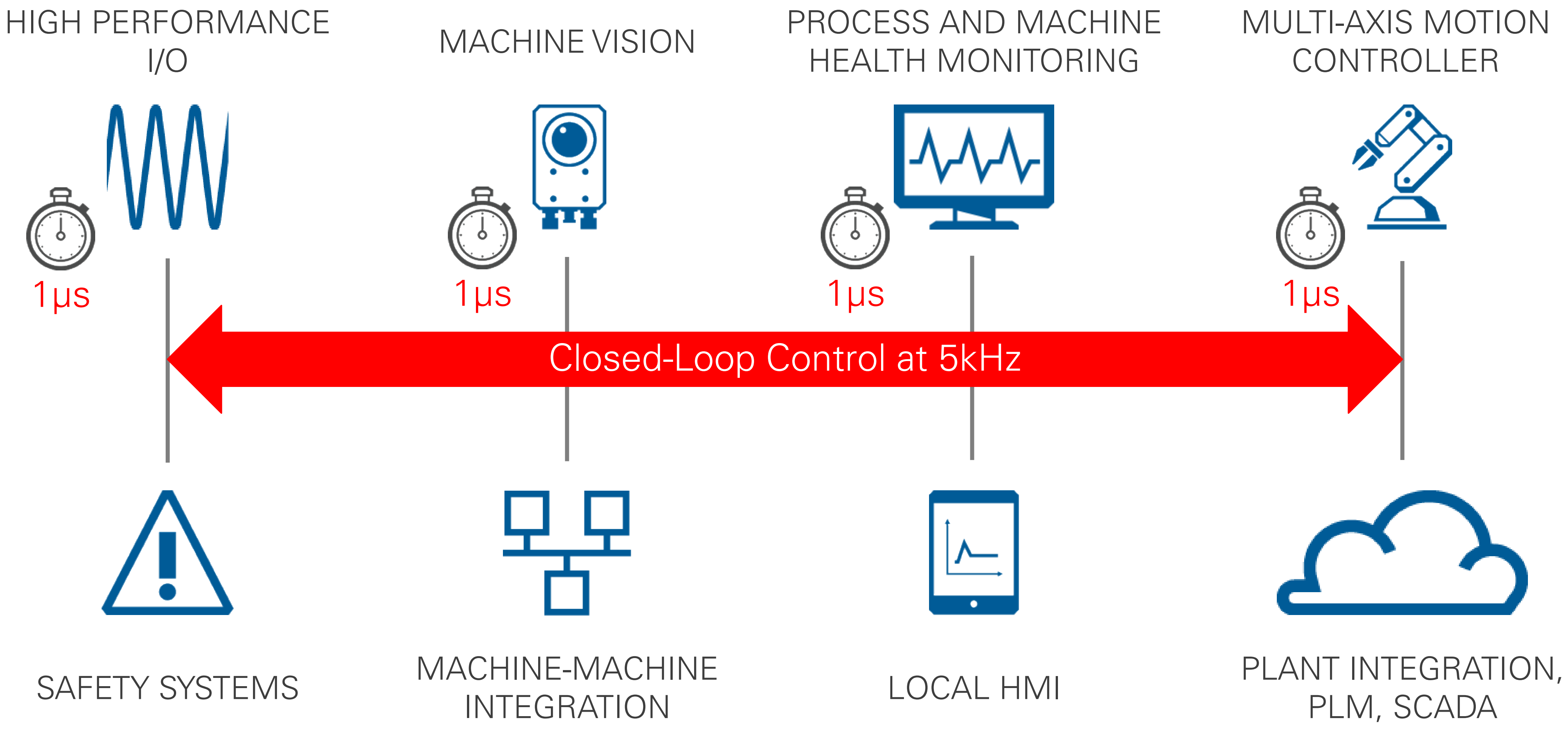
Instrument Design Libraries & Example Projects



NATIONAL INSTRUMENTS

Deterministic ethernet

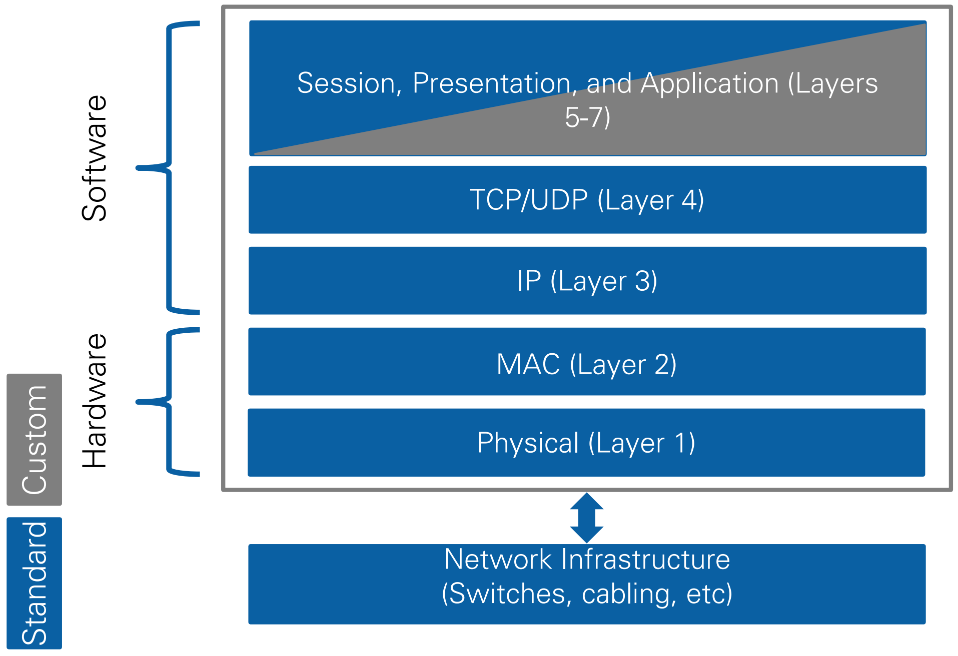
Typical machine control application sub-systems



Technical Needs of Communications

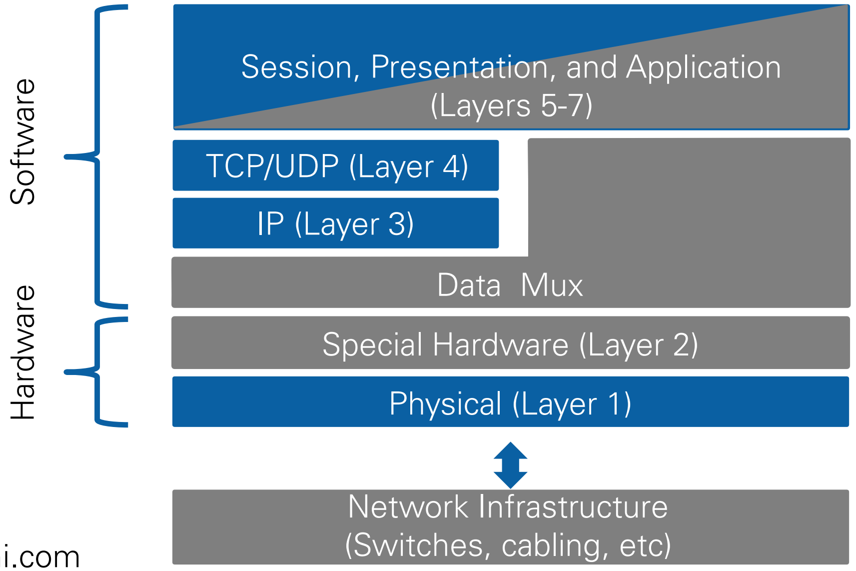
Feature	Need	Needed For
Guaranteed Bandwidth	Enable validation & analysis of system ability at design time	Reliable Operations
High Bandwidth	Enable high channel data and high speed streaming	Streaming of Data
Bounded Latency (and low)	Prioritize isochronous data over best effort on the same interconnect to maintain specified latency	Control Applications
Clock Synchronization	Allowing producers and consumers of isochronous data to be phase coordinated Allow Application synchronization	Synchronized IO and Distributed Control
Distance	Enable separation of IO from controller or measurements of physically large systems	Application Dependent
Topology	Provide physical options for wiring	Application Dependent
Ecosystem	Enable the inclusion of third party devices such as drives	Application Dependent

The Challenge



"Standard" Ethernet

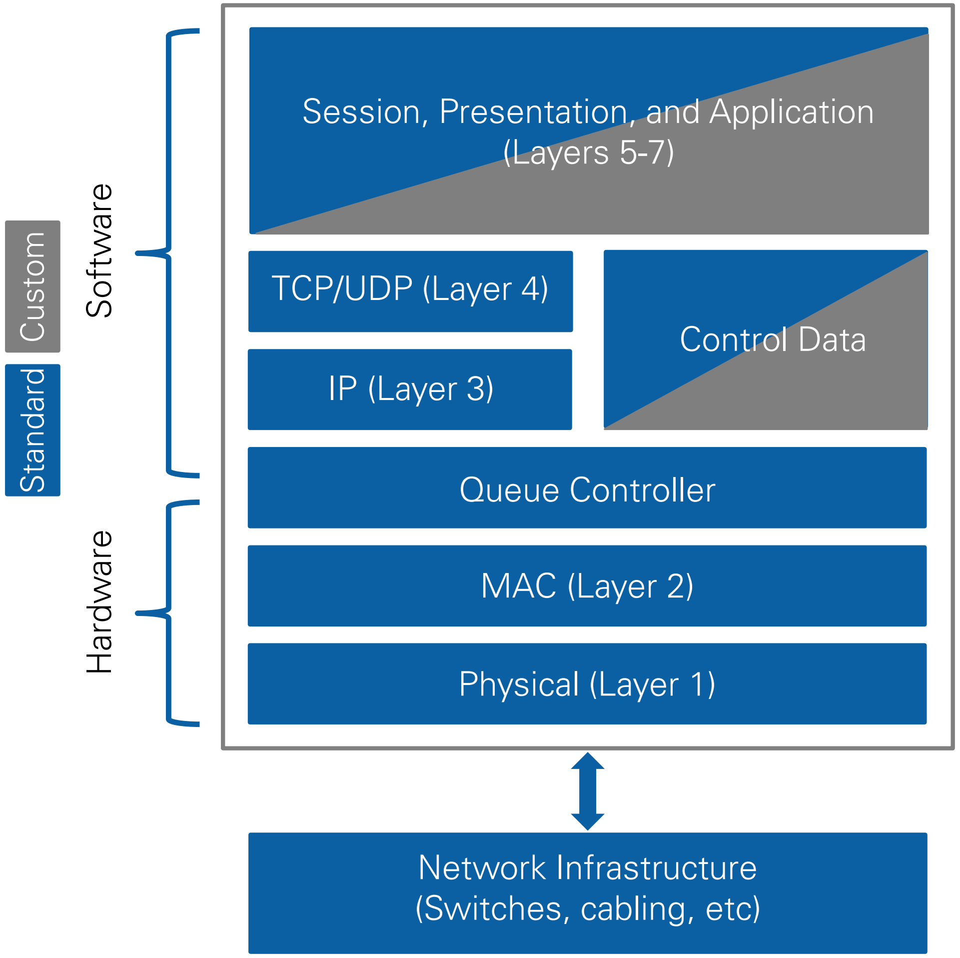
- Best-in-class approach for openness and interoperability
- Cannot bound latency (needed for control applications)
- Cannot guarantee bandwidth (needed for reliability)



"Hard Real-Time" Ethernet

- Best-in-class approach for latency and control
- Cannot "share the wire" (no third party devices)
- Cannot scale with Ethernet (e.g. limited to 100 Mb/s)
- Proprietary HW/SW increases costs

TSN-Based "Hard Real-Time" Ethernet Devices



TSN Ethernet

- Key industrial, embedded, and automotive vendors collaborating to drive requirements
- Best-in-class approach for control AND interoperability
- Bounded latency and guaranteed bandwidth
- Scales with Ethernet

NATIONAL INSTRUMENTS

Standards Efforts

Standards effort through IEEE 802 to improve latency and performance while maintaining interoperability and openness

Time Sensitive Networking (TSN) will provide:

- Time synchronization
- Bandwidth reservation for reliability
- Guaranteed bounded latency
- Low latency (preemption)
- Bandwidth (Gb+)
- Routable to support complex networks and wireless

Time Sensitive Networking: Key Elements

Time Synchronization

Traffic Scheduling

System Configuration

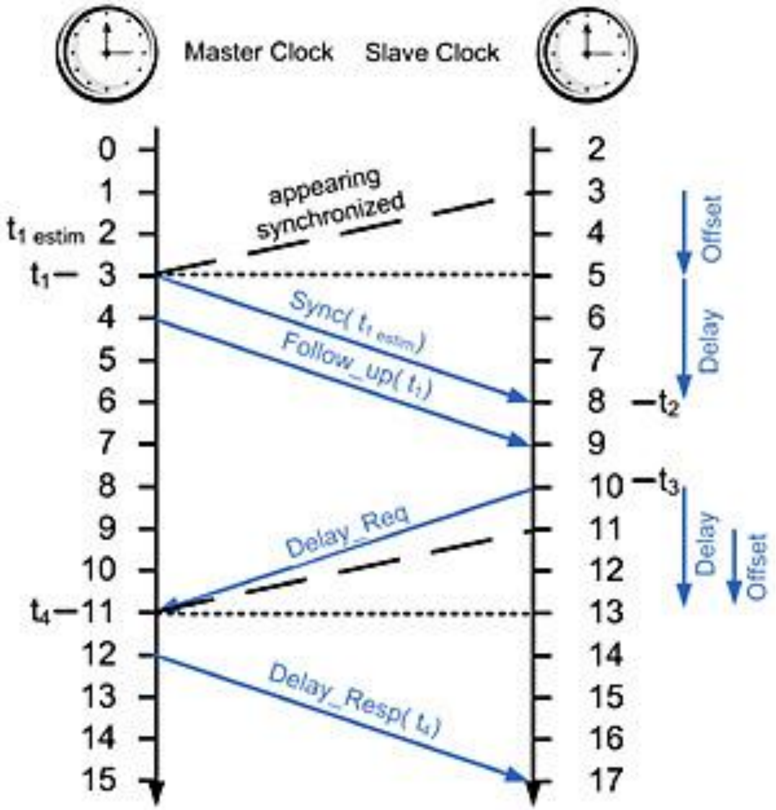
IEEE 802.1AS, IEEE 1588

Summary

End-nodes and switches share time

Features

- Synchronization of multiple systems using packet based communication
- Synchronization is possible over very long distances without impact from signal propagation delay



NATIONAL INSTRUMENTS™

IEEE 802.1Qbv

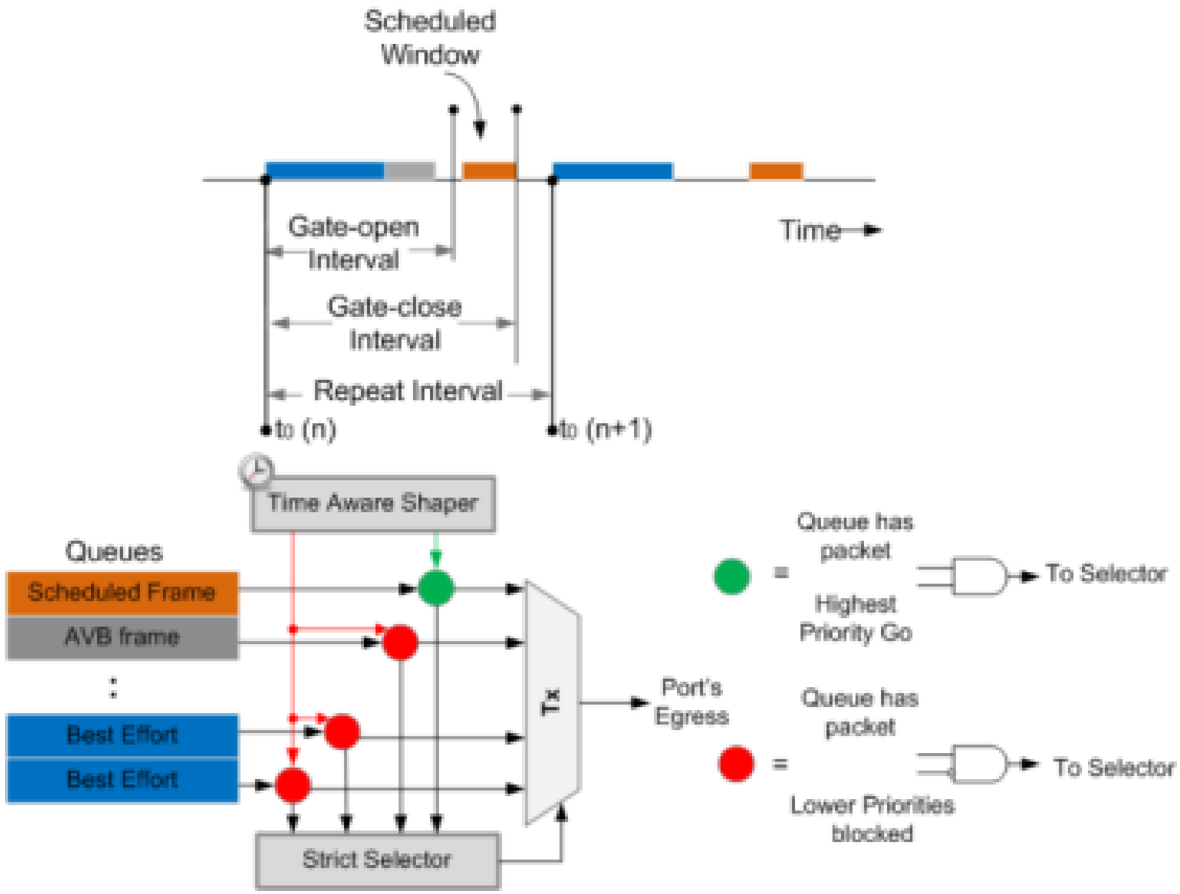
Traffic Scheduling

Summary

Every egress on the network is scheduled and follows a repeating cycle

Features

- Deterministic arrival of packets
- Scalable design with ability to assure multiple flows won't conflict



NATIONAL INSTRUMENTS

IEEE 802.1Qcc

Summary

Consistent mechanism for network configuration to meet the needs of end application

Features

- Standard mechanism for configuration of all network elements
- Configure “streams” between devices from any supplier

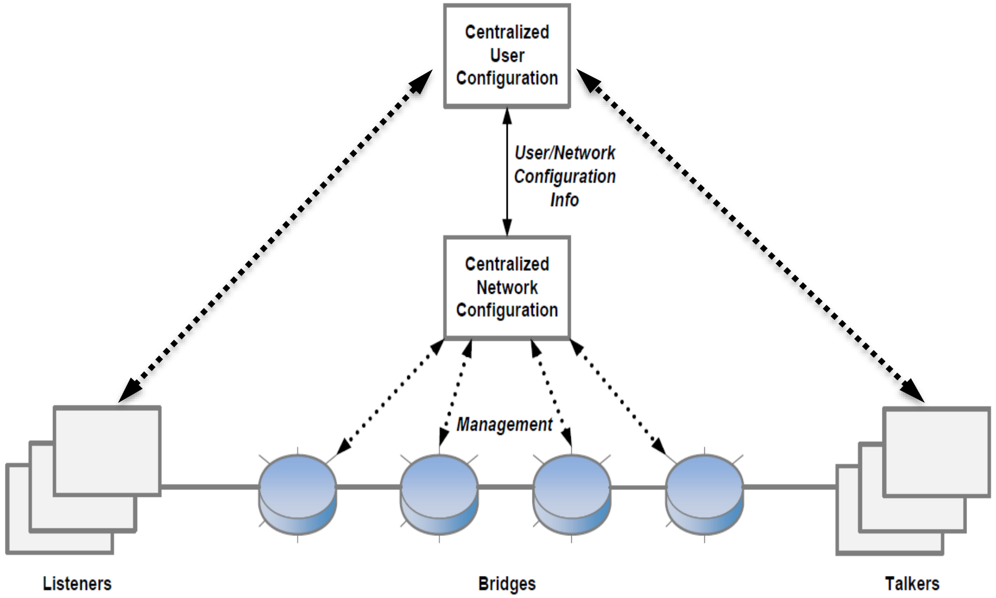


Figure 99-3 — Fully Centralized Model

National Instruments Investment

1. Time-based and isochronous programming in LabVIEW
2. Global time and synchronization for all processing elements and I/O
3. Bounded, low latency data transfer over Ethernet

CompactRIO

CompactDAQ

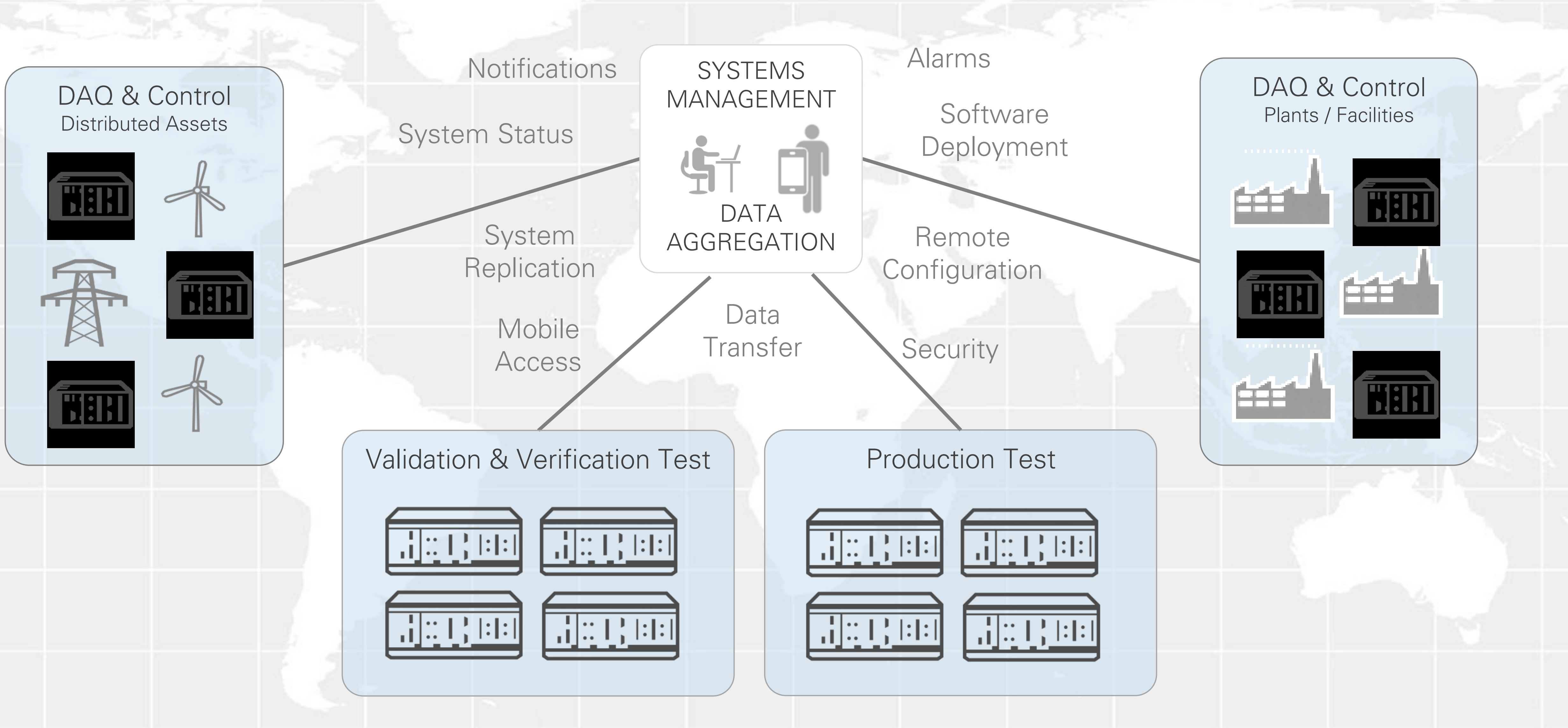
PXI

LabVIEW System Design Software

Manage deployed systems

Challenge: Managing Deployed Systems

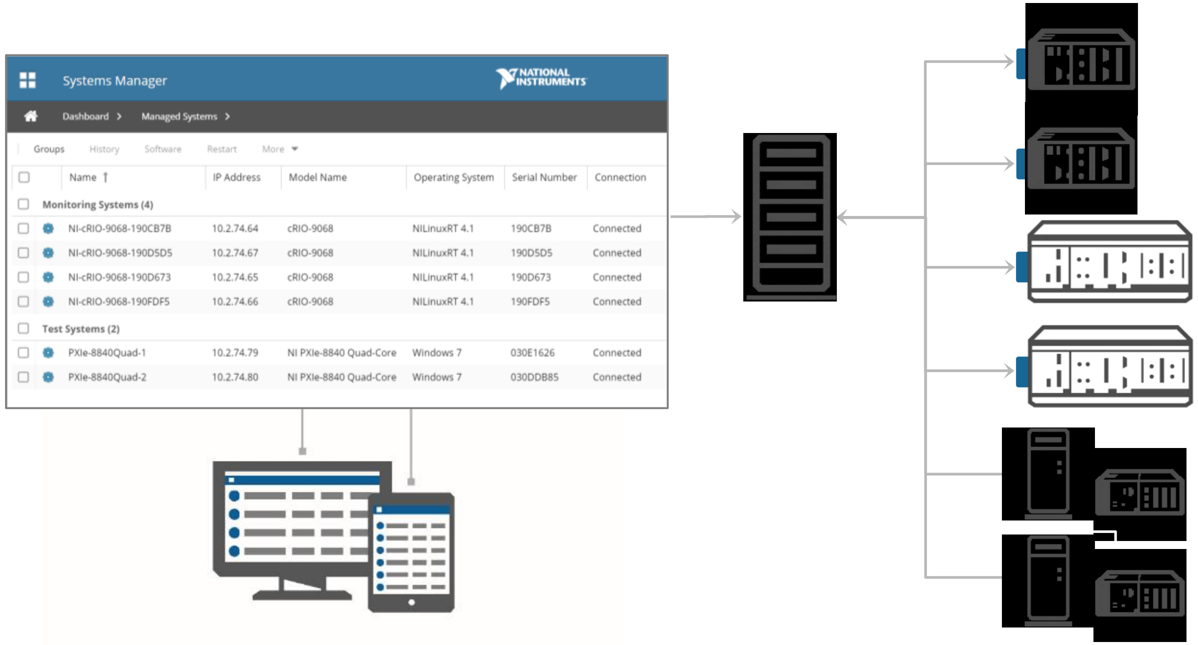
Customers with distributed systems often encounter challenges in the management of devices, software, and data



SystemLink

EARLY ACCESS RELEASE

Manage distributed systems with web application software that enables mass software deployment, device management, and data communications.



- Track and manage a group of connected systems
- Application access via web browser or mobile
- Install on-premise or with a cloud service provider

PRODUCT FEATURES

DEVICE MANAGEMENT

Register hardware targets and classify systems through a shared interface.

SOFTWARE CONFIGURATION

Deploy software to multiple remote targets, with upgrade, downgrade, & uninstall.

New module available in early 2018

SYSTEM HEALTH MANAGEMENT

Track and manage health of hardware with alarms and notifications.

DATA COMMUNICATIONS

Automate data transfer among connected nodes, using LabVIEW or Web APIs.

NATIONAL INSTRUMENTS

Device Management

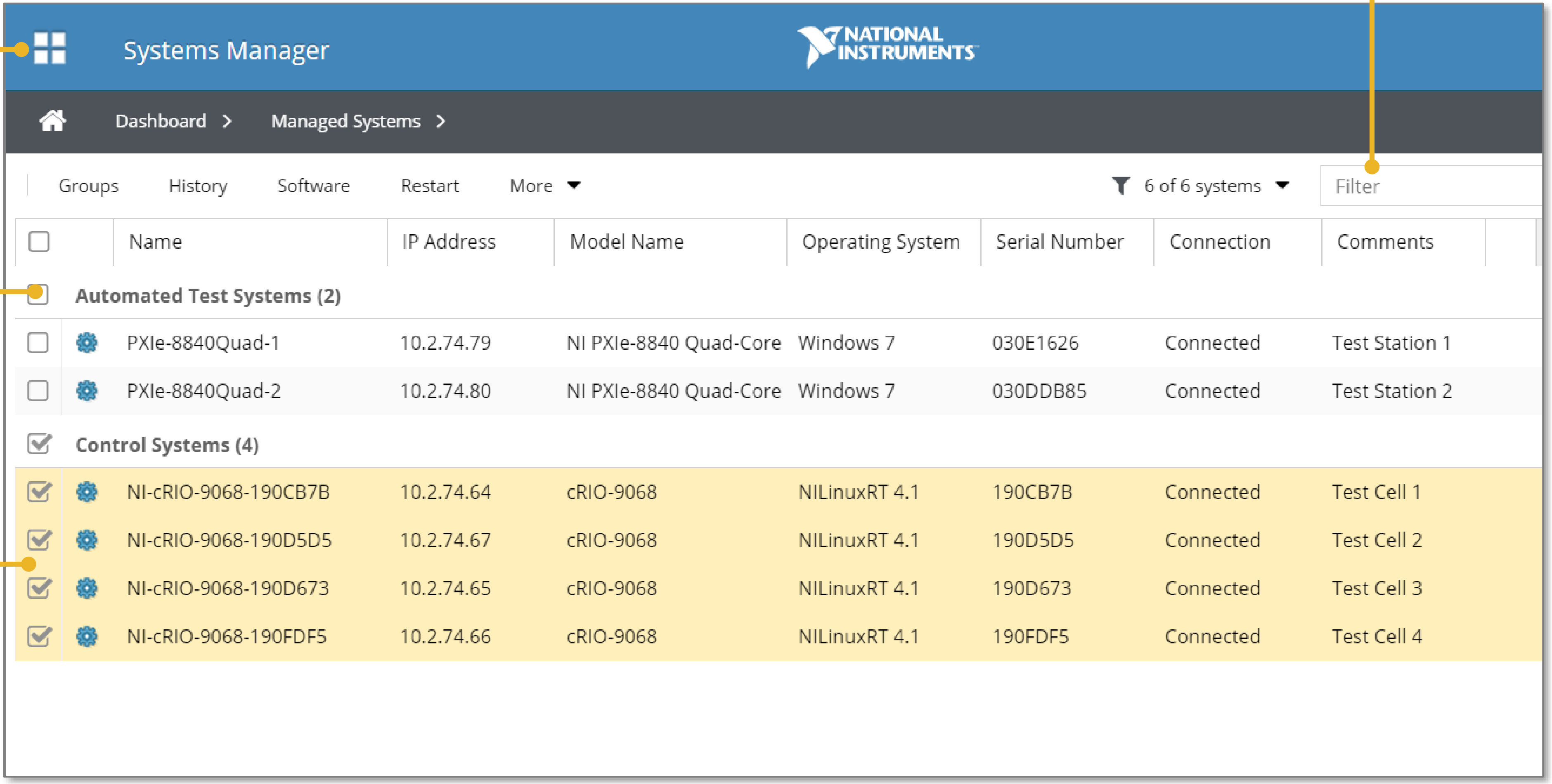
View and manage detailed system and device information

Track systems through a central web interface

Classify systems into groups

Multi-select to perform remote functions in parallel (e.g. install, restart).

Search systems across all groups

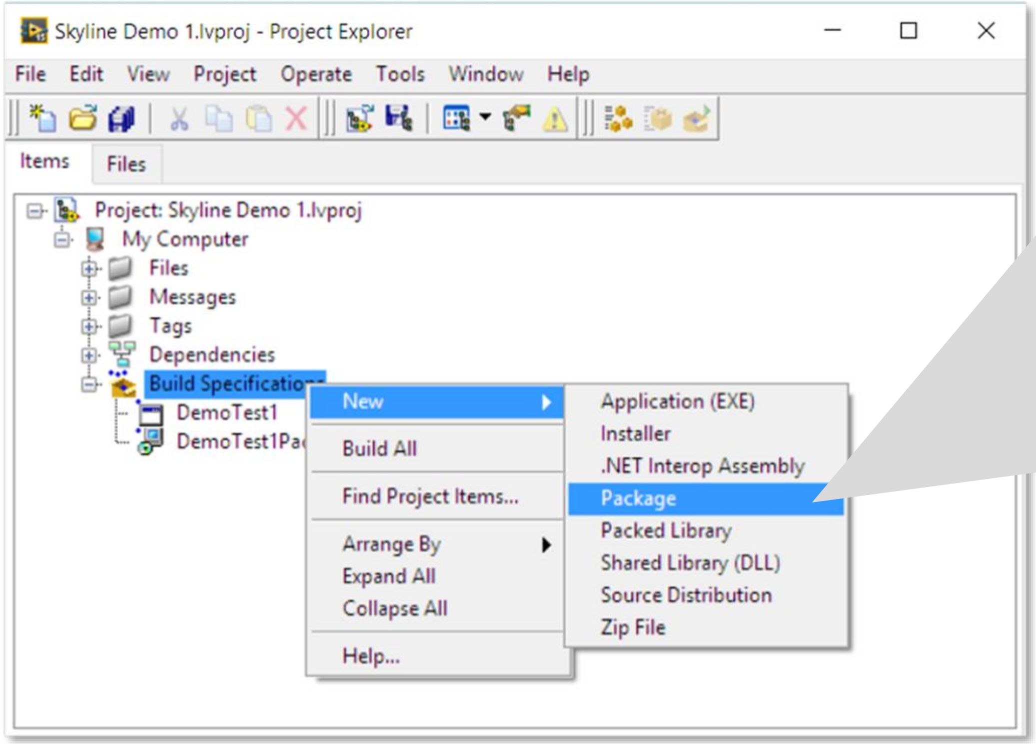


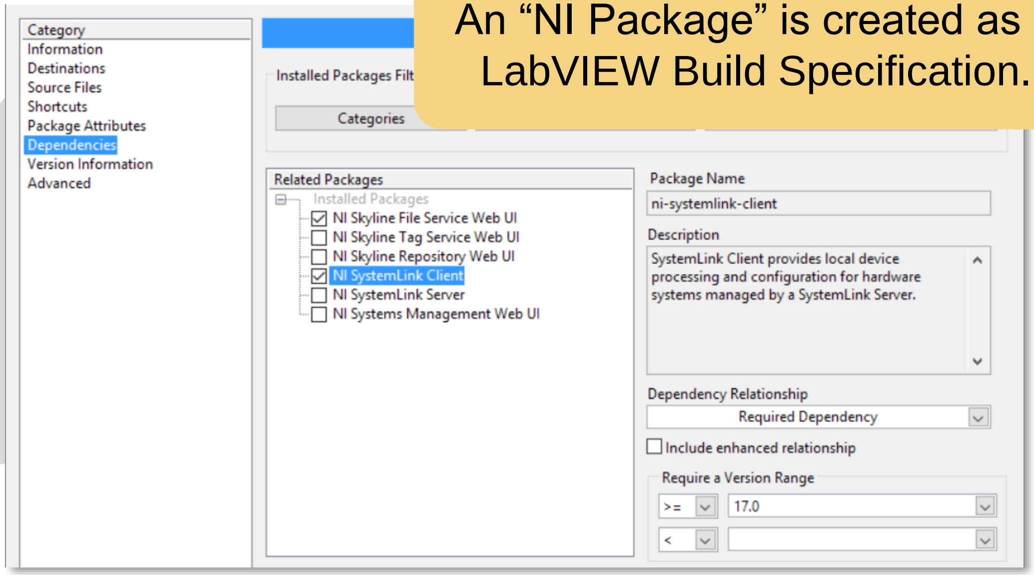
Name	IP Address	Model Name	Operating System	Serial Number	Connection	Comments
Automated Test Systems (2)						
PXIe-8840Quad-1	10.2.74.79	NI PXIe-8840 Quad-Core	Windows 7	030E1626	Connected	Test Station 1
PXIe-8840Quad-2	10.2.74.80	NI PXIe-8840 Quad-Core	Windows 7	030DDB85	Connected	Test Station 2
Control Systems (4)						
NI-cRIO-9068-190CB7B	10.2.74.64	cRIO-9068	NILinuxRT 4.1	190CB7B	Connected	Test Cell 1
NI-cRIO-9068-190D5D5	10.2.74.67	cRIO-9068	NILinuxRT 4.1	190D5D5	Connected	Test Cell 2
NI-cRIO-9068-190D673	10.2.74.65	cRIO-9068	NILinuxRT 4.1	190D673	Connected	Test Cell 3
NI-cRIO-9068-190FDF5	10.2.74.66	cRIO-9068	NILinuxRT 4.1	190FDF5	Connected	Test Cell 4

Software Deployment

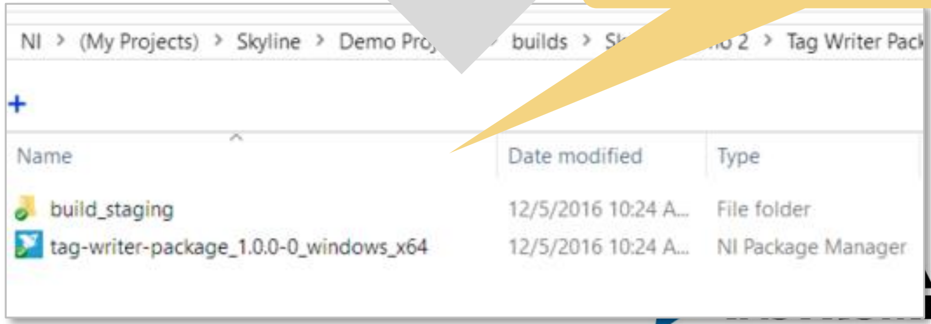
CREATE PACKAGES → UPLOAD TO REPOSITORY → DEPLOY !

Package creation from LabVIEW:





An "NI Package" is created as a LabVIEW Build Specification.



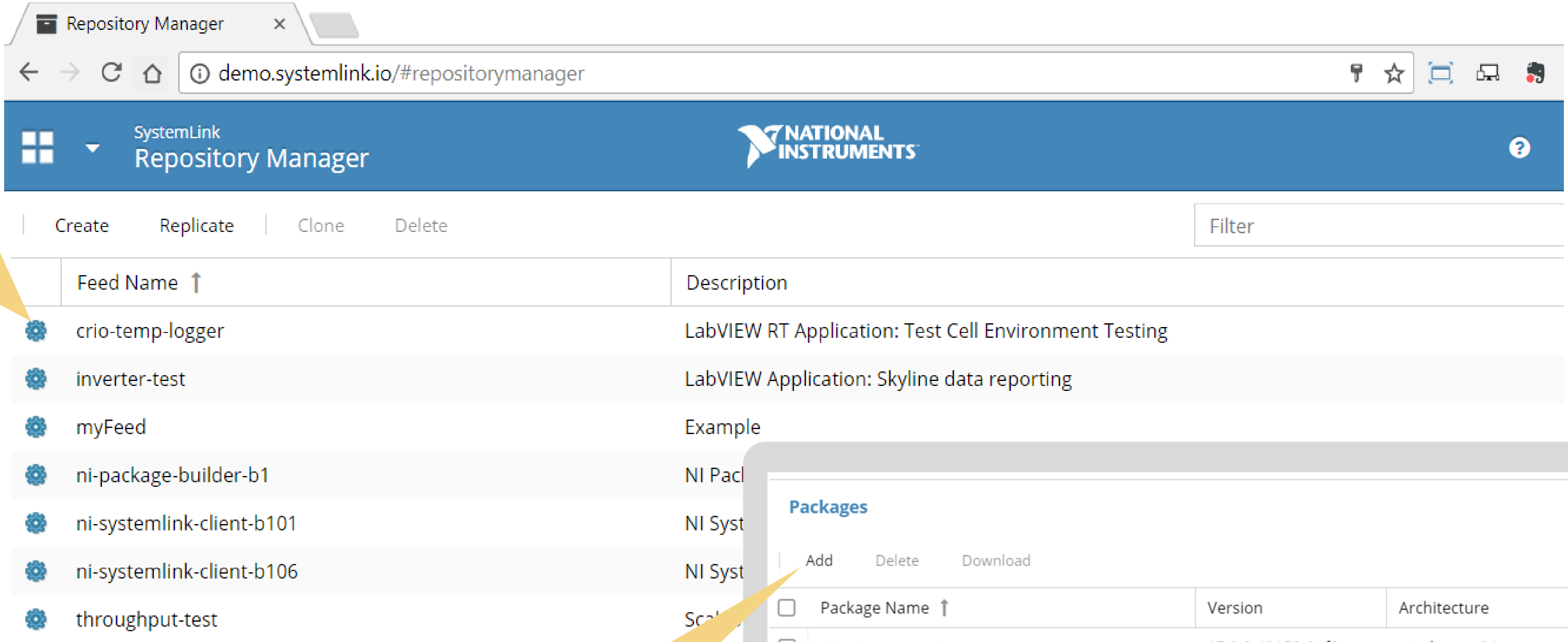
NI Package file on disk.

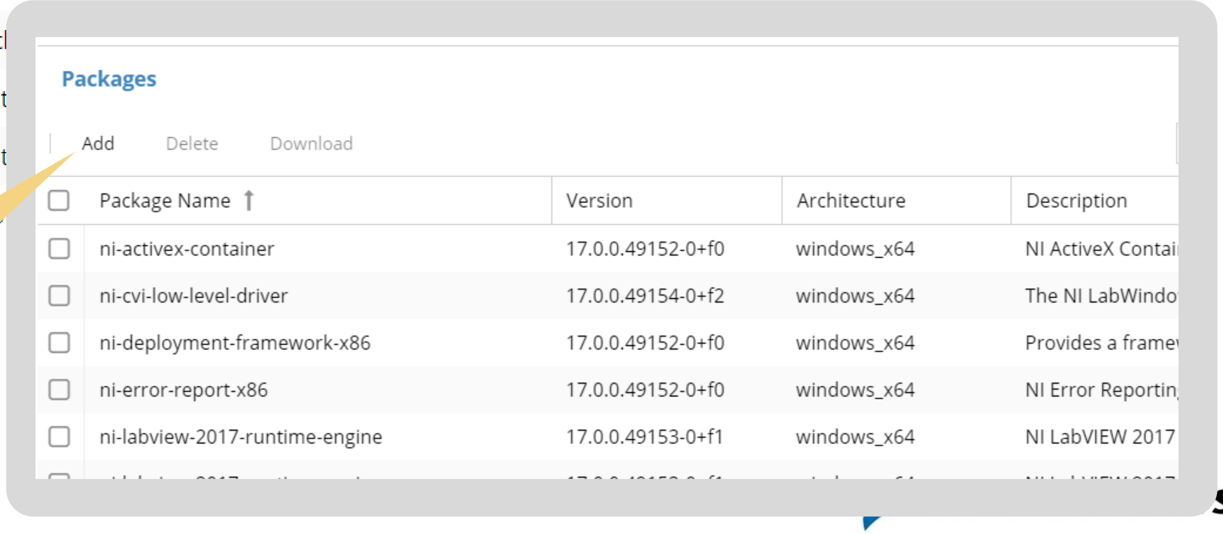
A command-line interface is also available for package creation.

Software Deployment

CREATE PACKAGES → **UPLOAD TO REPOSITORY** → DEPLOY !

Create deployment Feeds to organize packages.



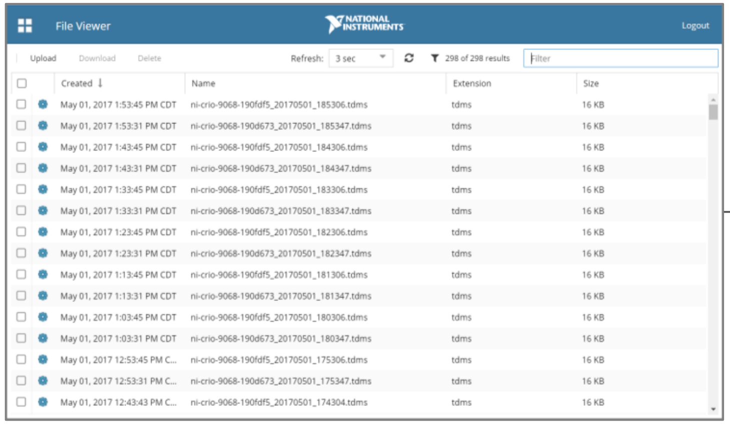


Easily add new packages to a Feed.

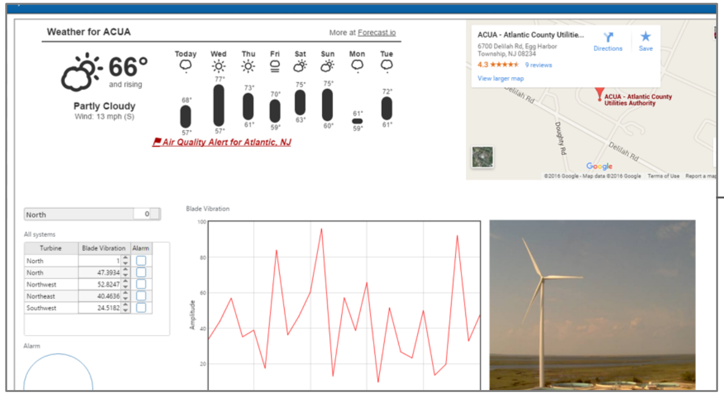
Data Communication

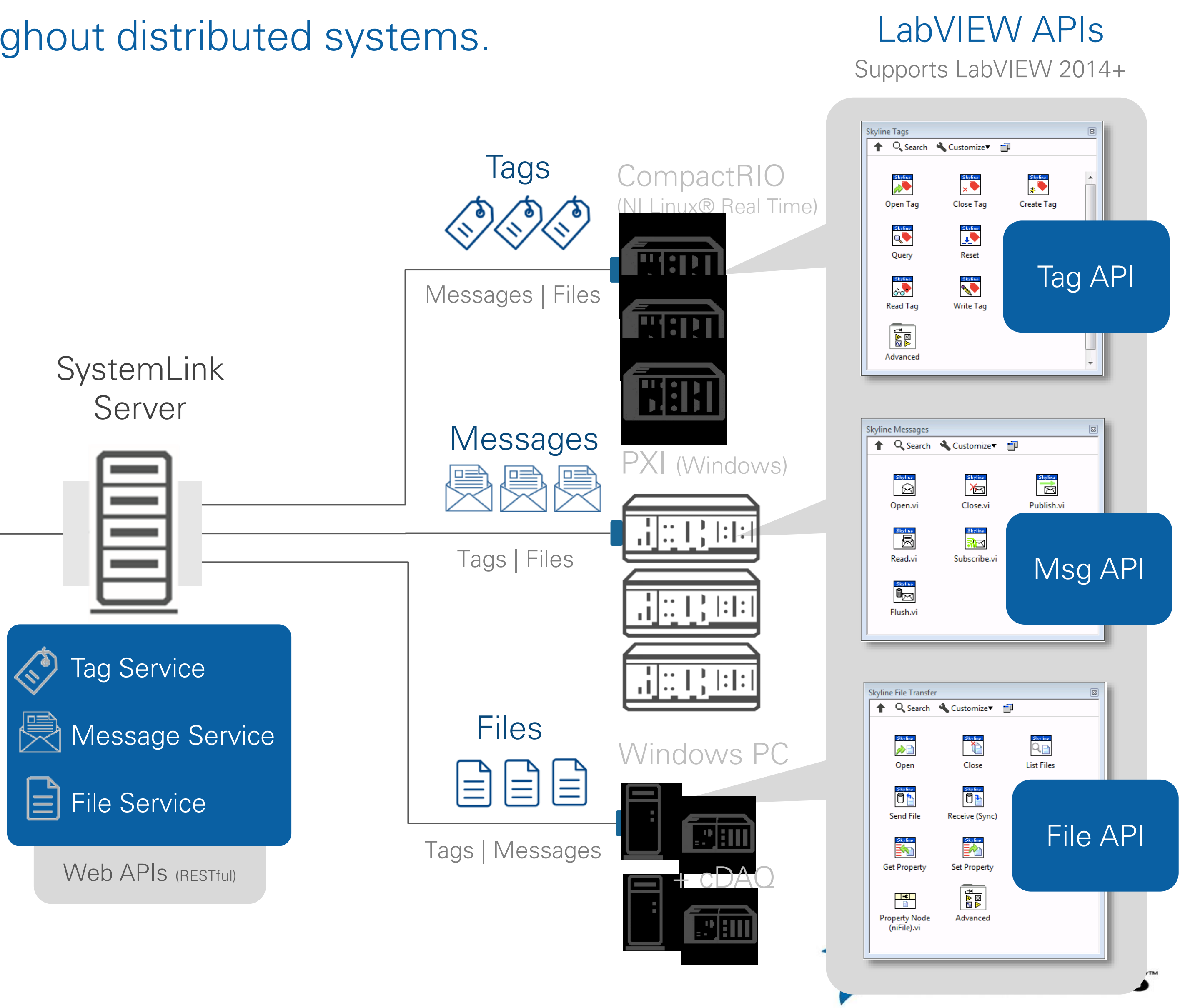
Automate and manage data transfer throughout distributed systems.

Data Viewers for Files & Tags

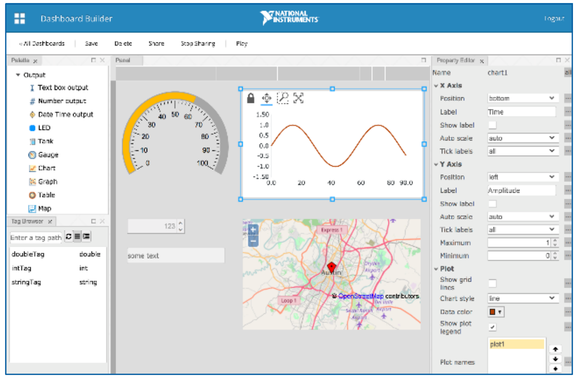


Custom Web Applications



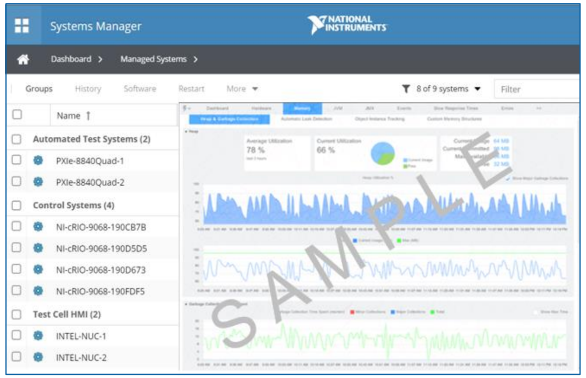


Roadmap: Early 2018 Features



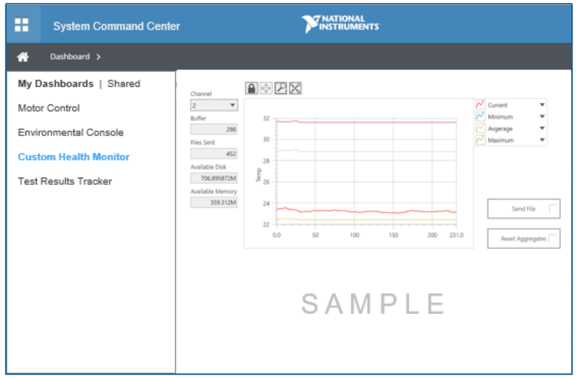
Dashboard Builder

- Browser-based application editor
- Drag-and-drop visualization widgets
- *No coding required*
- Connect UI controls to tags
- Mobile layouts



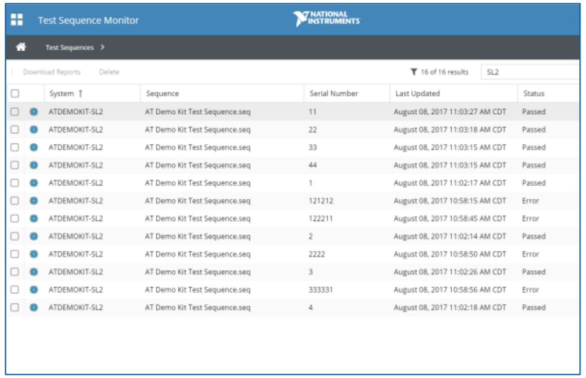
System Health Monitoring

- Monitor health metrics
- Configure alarm services
- Manage triggers and notifications
- Extensible with LabVIEW and Web APIs



WebVI Hosting

- Create web UIs in LabVIEW NXG 2.0
- Host WebVIs on SystemLink Server
- VI executes in the browser (no plug-ins)
- Incorporates block diagram architecture
- Leverage data from tag service



Test Sequence Monitor

- Central display of test status
- Interface with test executive
- TestStand plug-in
- Live updates as tests execute
- View/download test reports

SystemLink Architecture

Open and Extensible

APIs for LabVIEW and Web Services

Leverages several open-source standards

Cloud-Ready

Install on premise or in the cloud

Cloud-centric developments on the horizon

Optimized for Distributed Nodes

Systems functions execute in parallel

Targets, server, & users can be on multiple networks

Secure

Data communications are encrypted via TLS

Secure user access: LDAP & AD integration

SystemLink: Supported Software & Hardware

Deployment Software

Native Packaging Tools

LabVIEW* (32 & 64-bit)	2014-2017
LabVIEW Real-Time	2017
LabVIEW NXG	2.0
TestStand	2017

* A LabVIEW plug-in is available to create deployment packages with EXEs, PPLs and Source Distributions.

Command-Line Support

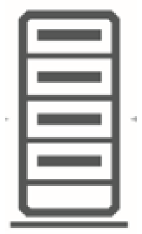
NI & Non-NI Software

- Applications, libraries, drivers, docs, installers, etc. can be packaged using command-line interface and deployed with SystemLink

Hardware

SystemLink Server

WINDOWS OS Options: 64-bit



Windows Server

- Windows Server 2008 R2, SP1
- Windows Server 2012 R2
- Windows Server 2016

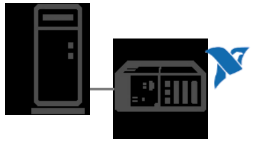
Windows PC

- Windows 7+

SystemLink Clients

WINDOWS TARGETS: Win 7+, 64-bit

NI PXI

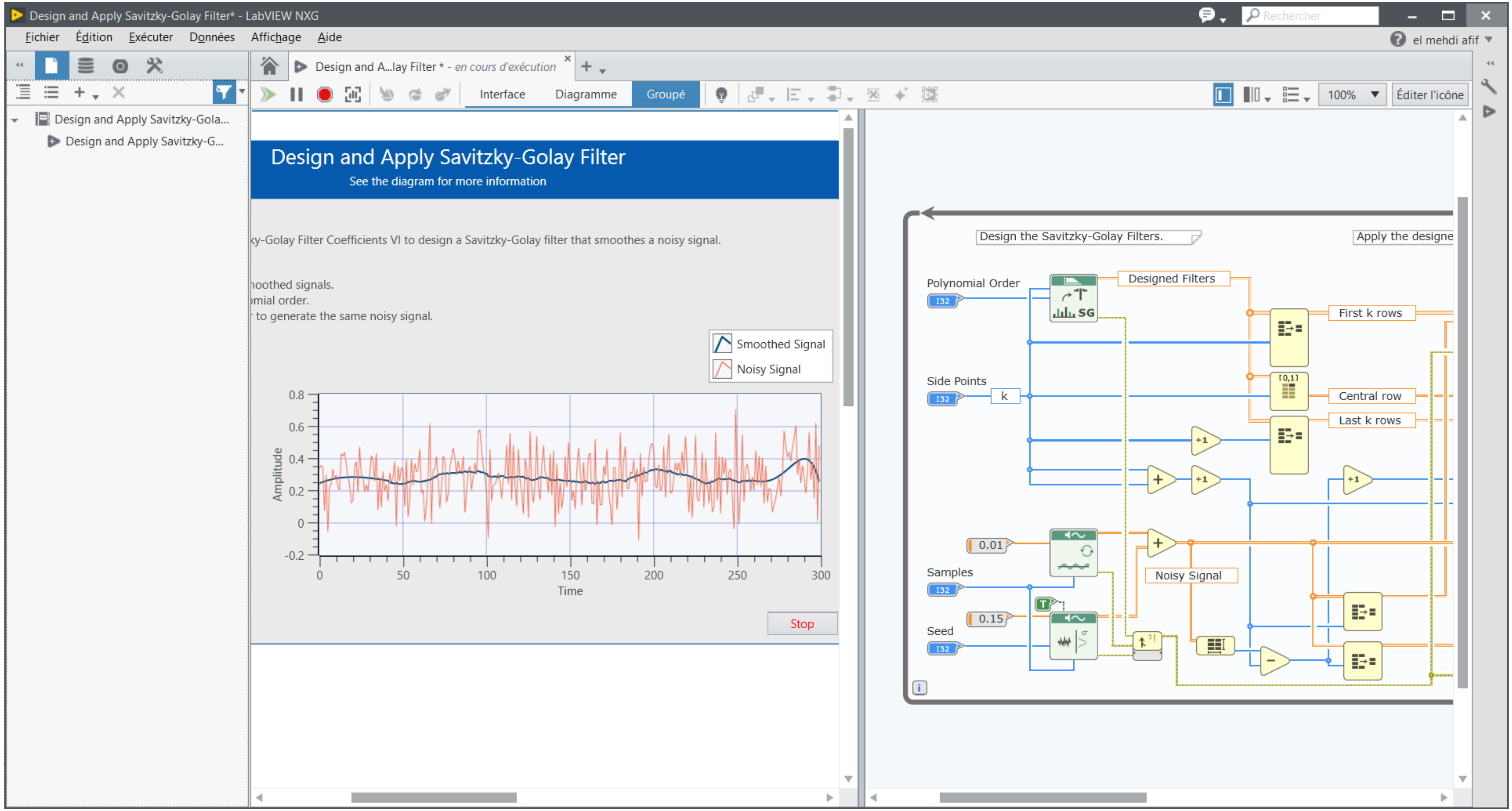


Windows PC

NI LINUX RT TARGETS

NI CompactRIO

LV NXG

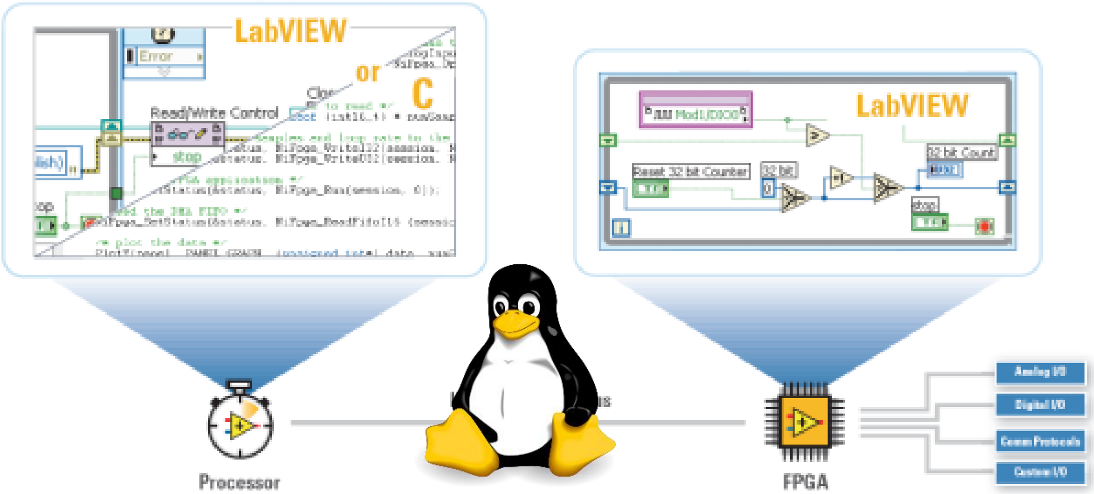

Design and Apply Savitzky-Golay Filter* - LabVIEW NXG
Fichier Édition Exécuter Données Affichage Aide
el mehdi afif
Design and A...lay Filter * - en cours d'exécution
Interface Diagramme Groupé
100%
Éditer l'icône
Design and Apply Savitzky-Gola...
Design and Apply Savitzky-G...
Design and Apply Savitzky-Golay Filter
See the diagram for more information
ky-Golay Filter Coefficients VI to design a Savitzky-Golay filter that smoothes a noisy signal.
noothed signals.
mial order.
to generate the same noisy signal.
Smoothed Signal
Noisy Signal
Amplitude
Time
Stop
Design the Savitzky-Golay Filters.
Apply the designe
Polynomial Order
Designed Filters
First k rows
Side Points
Central row
Last k rows
Samples
Noisy Signal
Seed
0.01
0.15

NI Linux RT, Linux Desktop

Supported Hardware

LabVIEW Support and Freedom in Development.

- Enjoy the **flexibility** of Linux, with the **determinism and reliability** of a real-time operating system.
  - Desktop UI, Peripherals, System Administration, Real-Time schedulers
- Leverage the vast **ecosystem** of tools and IP
  - Networking, Configuration Management, Simulation, Monitoring, etc.
- **Reuse** C/C++ code in and alongside LabVIEW Real-Time built applications
  - FPGA Interface C API, System Configuration C API



NATIONAL INSTRUMENTS

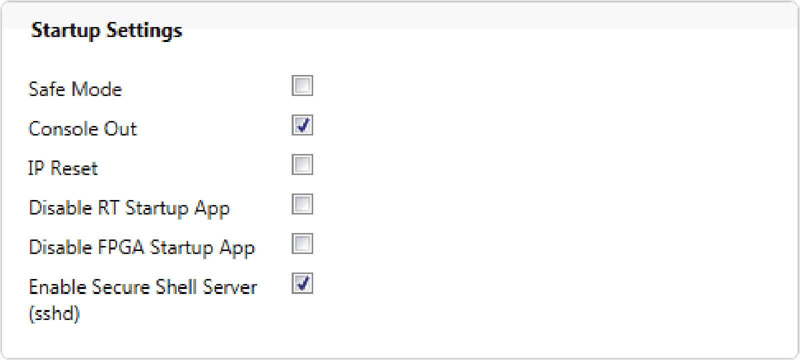
Secure Shell (SSH)

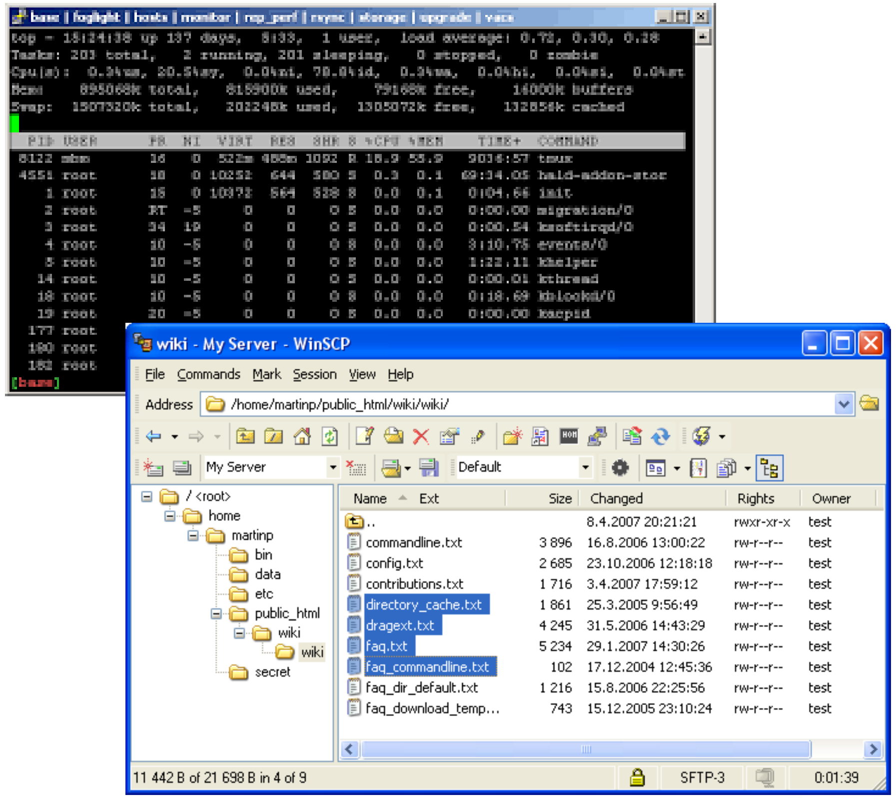
- Enable through MAX and/or Web Interface

- Can be used as a console

- Can be used to transfer files
  - Permissions based on login
  - SFTP

- Credentials synchronized with NI-Auth (Web Interface)





NATIONAL INSTRUMENTS

Leveraging the Linux Community

Databases	Security	Code Reuse	Connectivity
SQLite	IPTables	C/C++	NTP
MySQL	OpenSSL	Shell Scripts	SNMP
PostgreSQL	OpenVPN	Python	IPv6

- NI Package Repository: download.ni.com/ni-linux-rt/
- OS source: github.com/ni
- Kernel Driver Support

Security on NI Linux Real-Time

- SSL enabled by default
  - Can programmatically install software over SSL
  - Can use public keys for SSH

- IPtables* available for setting up a firewall

- OpenVPN* available for setting up a VPN

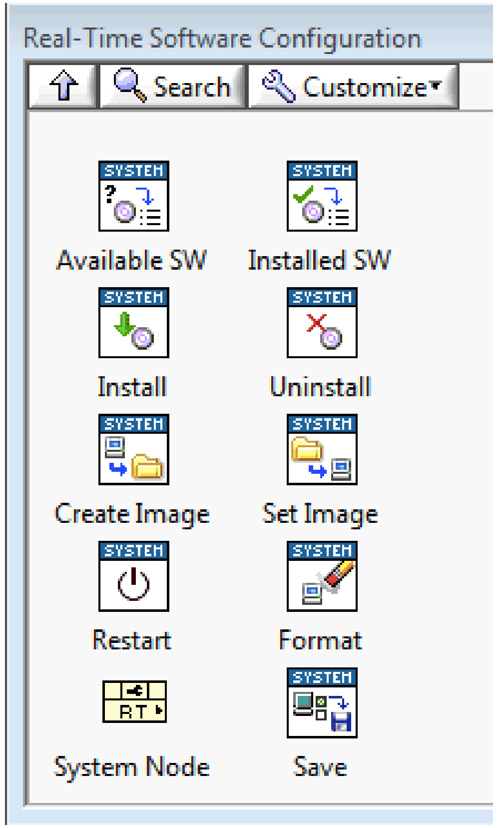
OPENVPN™

*Not supported by Applications Engineering. Requires experience. No LabVIEW API

NATIONAL INSTRUMENTS™

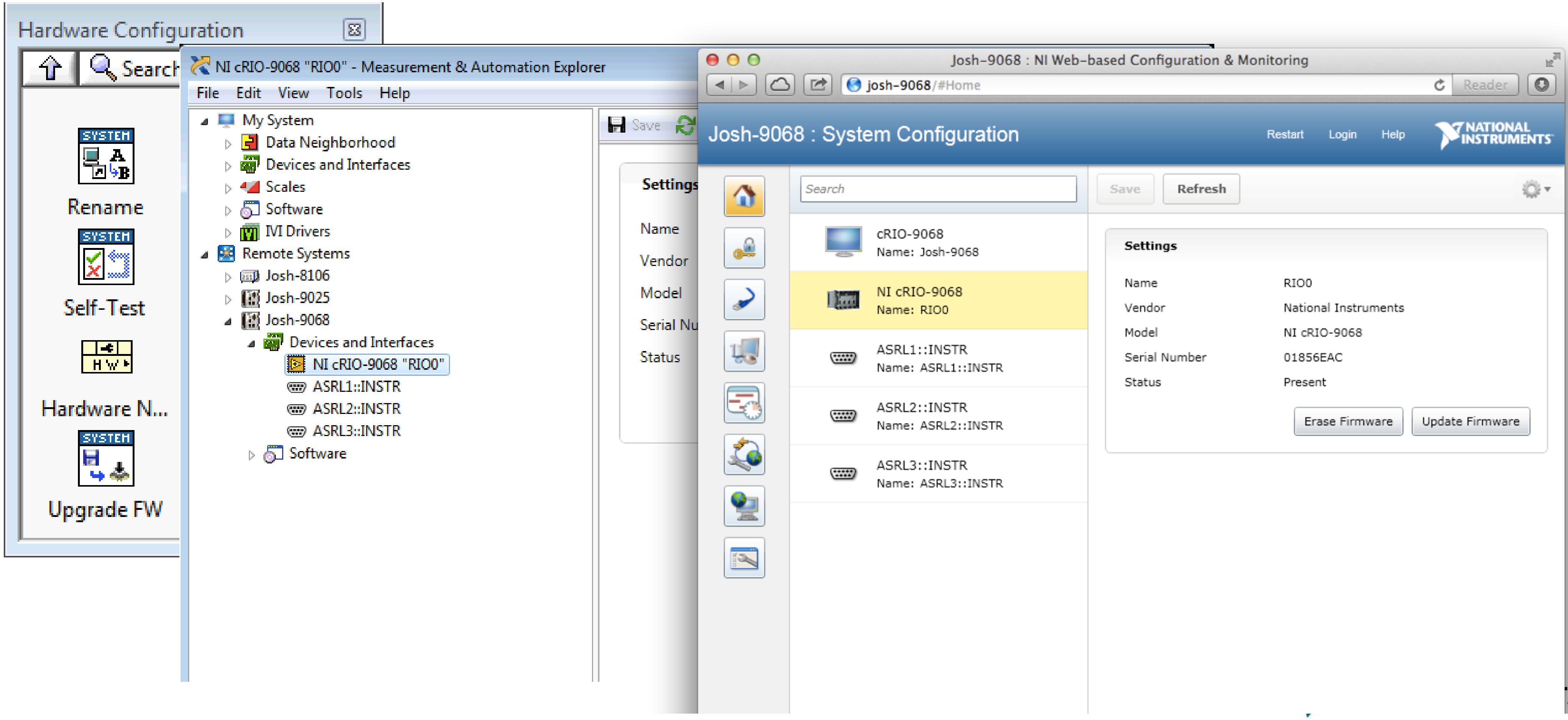
System Updates on NI Linux Real Time

- NI Linux Real-Time targets can directly call "Set Image"
  - Enables targets to reimage themselves
  - Images can be pulled down from the network or stored on a USB drive
- Specify additional metadata when creating an RT image (title, version, description)



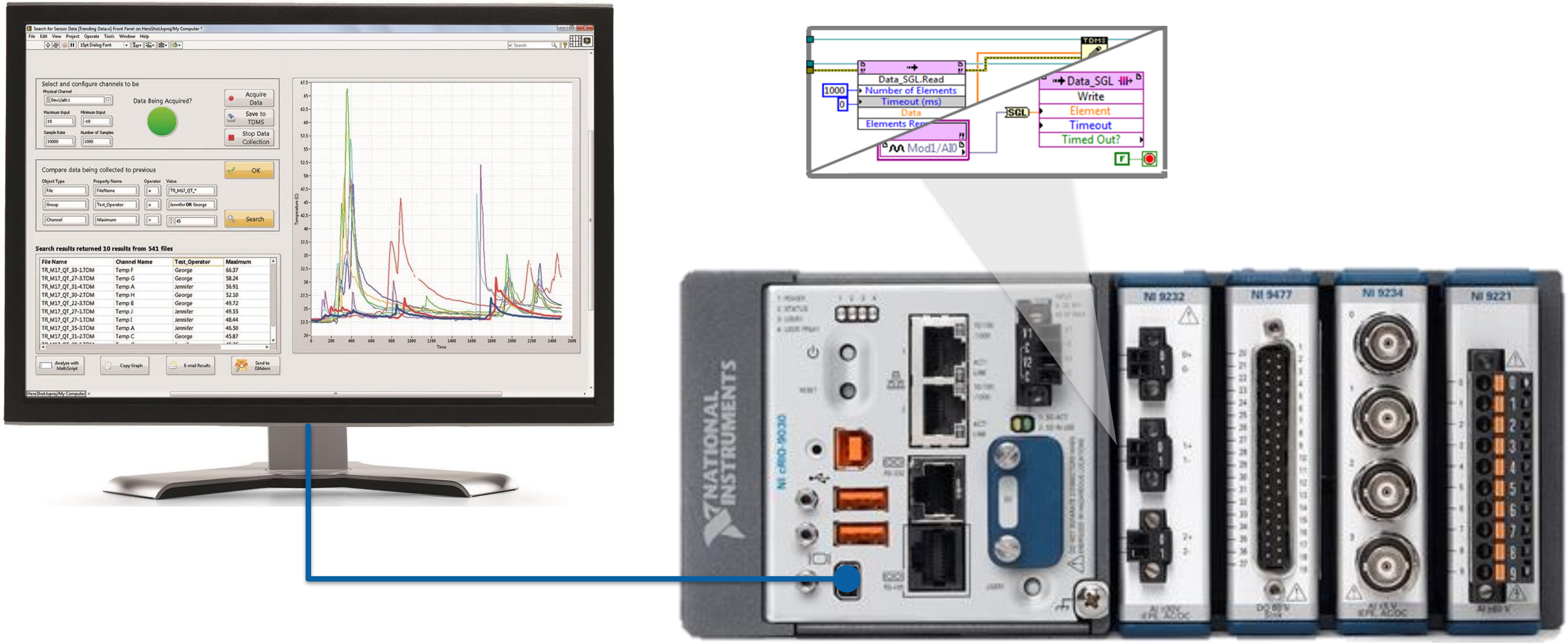
Manage FPGA Bit Files

- Update and erase the FPGA bit files on NI Linux Real-Time targets programmatically, from MAX, and the web



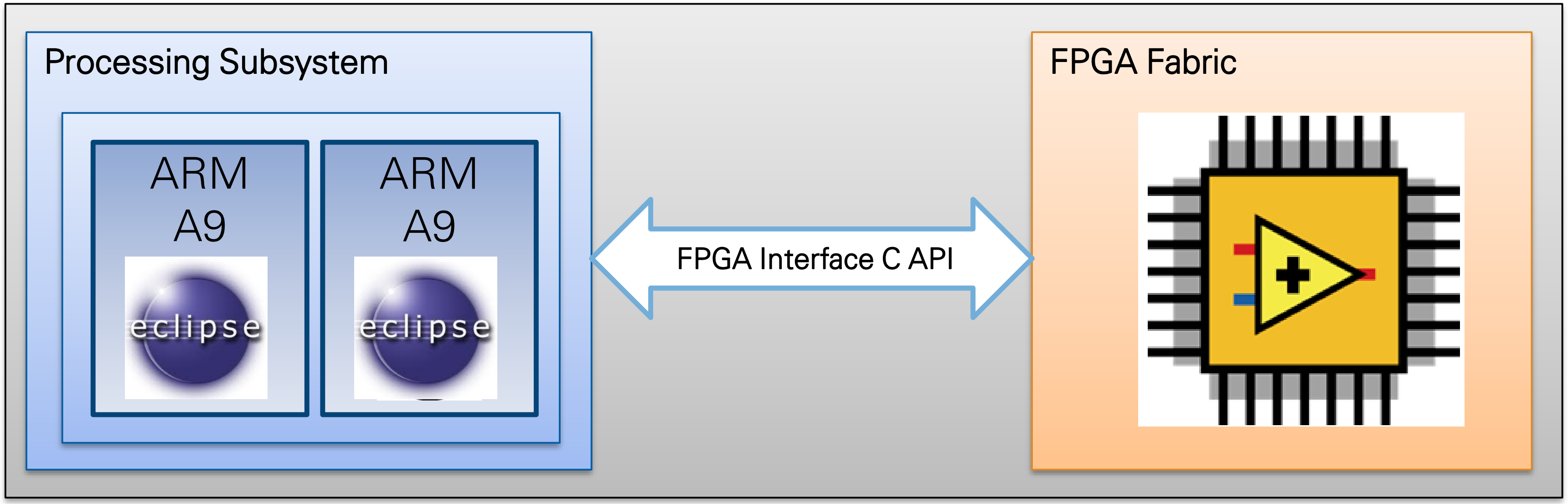
LabVIEW 2014 Real-Time with Embedded UI

Simplify system complexity by implementing a local HMI on the cRIO

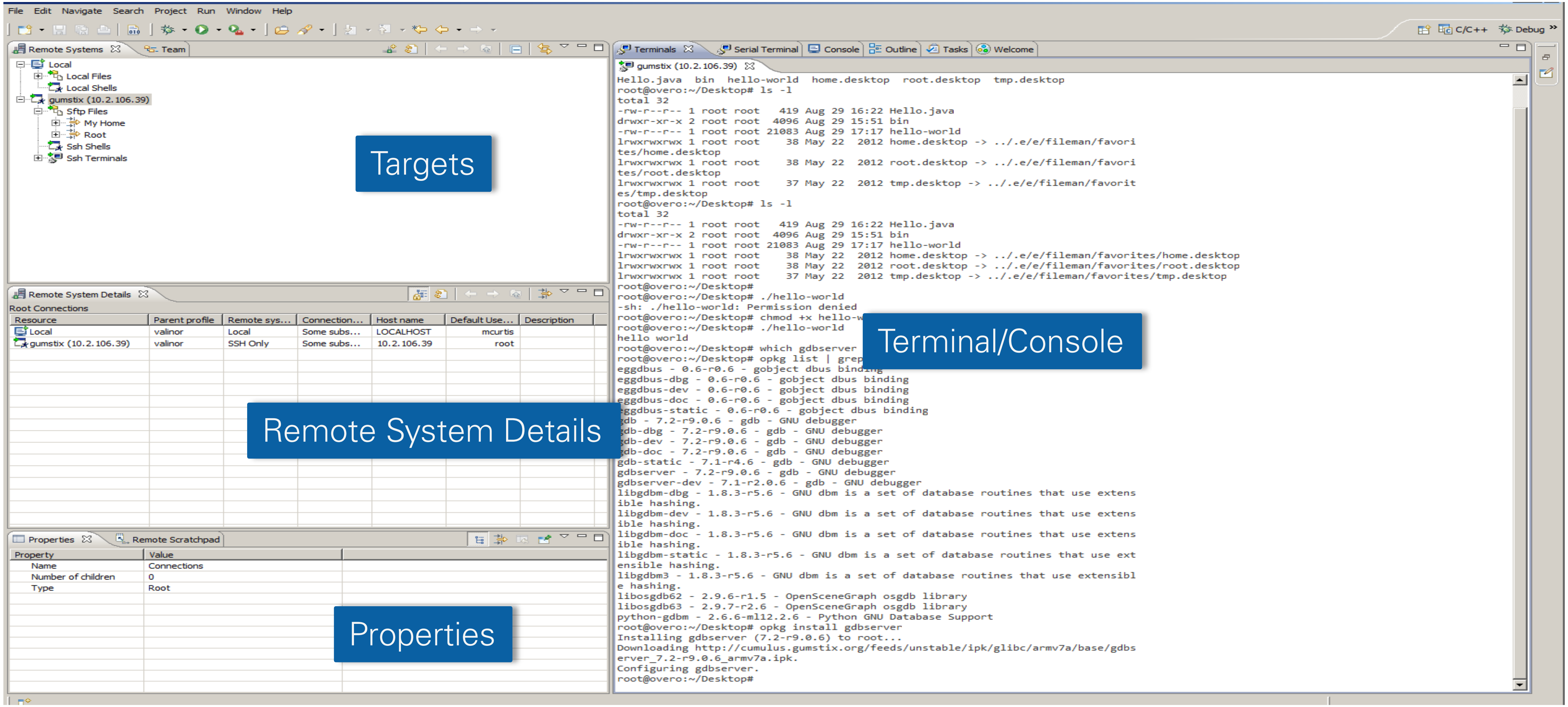


Eclipse for CompactRIO

- Choice of C and/or LabVIEW for programming processor
  - LabVIEW FPGA still required
  - FPGA Interface C API provides access to the FPGA from C
- Installer provided that includes Eclipse and Compiler
  - Available on ni.com/downloads

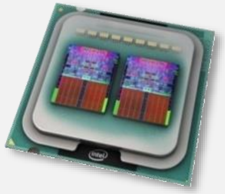


Eclipse Remote System Explorer



PXI Multi-Processing

Different NI supported Processing Technologies

Technology	Advantages
 Multi-Core CPUs	•Floating Point Operations •Diversity of Tasks •Parallelism based on n-cores
FPGAs	•Direct Connection to I/O for In-Line Processing •High Parallelism •High Throughput (fixed-point operations)
GP-GPUs	•Potentially High Throughput and Parallelism (for specific tasks)

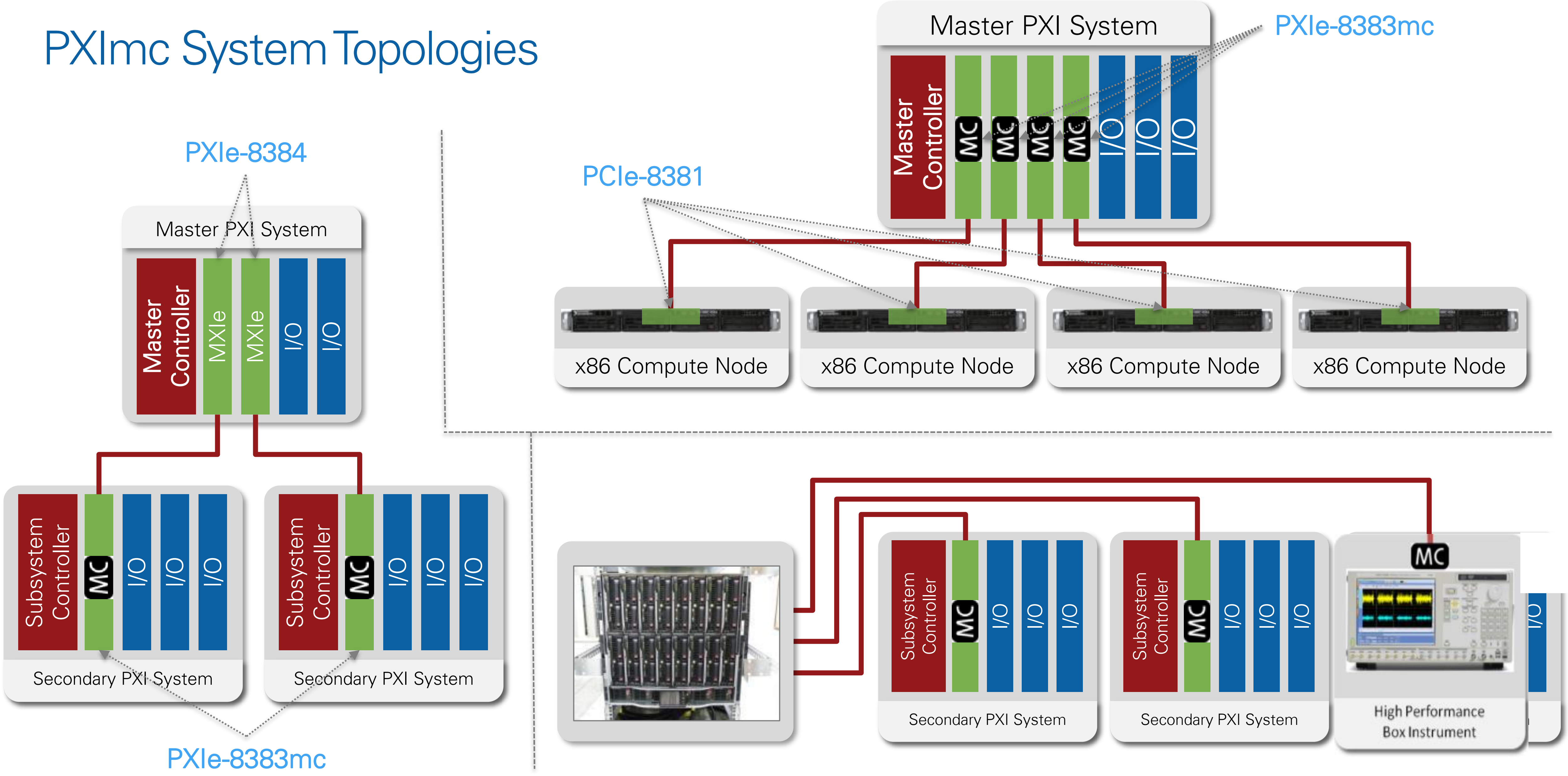
But we need high-bandwidth and low-latency data transfer for distributed PXI chassis

Comparing Buses for Data Transfer

- Comparison Vectors:
  - **Bandwidth:** The amount of data that can be transmitted in a given time
  - **Latency:** The time it takes from the first bit to travel from the transmitter to the receiver.

	Gigabit Ethernet	10 Gigabit Ethernet	PCI Express	Reflective Memory
Bandwidth	Good (60-70MB/s)	Better (600-700MB/s)	Best (3 GB/s)	Good (170 MB/s)
Latency	Good (mS range)	Good (mS range)	Best (uS range)	Best (uS range)

PXImc System Topologies



NATIONAL INSTRUMENTS™

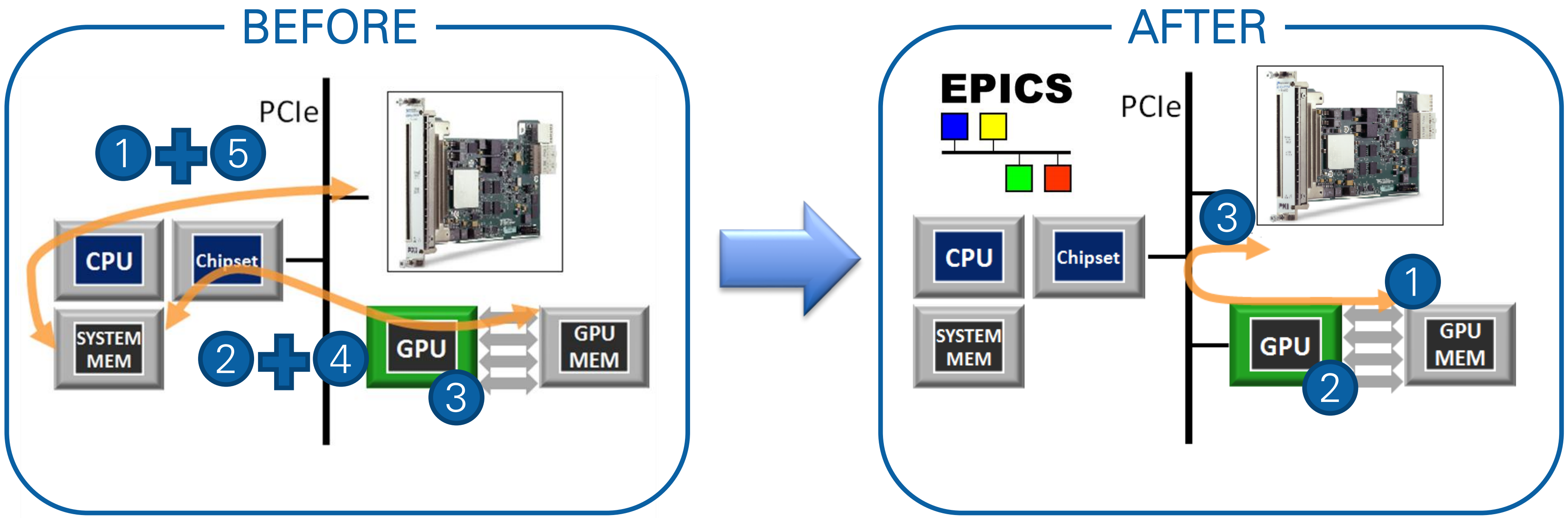
Co-Processing Module

Industry's First PXI Express Co-Processing Module

- Intel® Core i7-4700EQ processor
- 4 Physical and 8 Logical Cores
- 2 x USB 2.0, 1 x Gigabit Ethernet LAN ports
- 4 GB (1 x 4 GB DIMM) dual-channel 1600 MHz DDR3 RAM
- Up to 4GB/s theoretical (2.7 GB/s actual) bandwidth for data transfer (single direction)
- 5 micro-second total (SW+HW) latency between co-processing module and main CPU

GPU processing (UPM University)

- Leverage FlexRIO, P2P, GPU and NI-RIO Open Source to implement continuous real time DAQ&Processing systems **with minimum CPU intervention**.
- Develop standardized methodologies to integrate these technologies in scientific research environments using NDS-EPICS



NATIONAL INSTRUMENTS

Stay Connected During and After NIDays

ni.com/community

facebook.com/NationalInstruments

twitter.com/niglobal

youtube.com/nationalinstruments