

Hands-on Embedded Systems 2014

Developing Monitoring and Control Systems

with LabVIEW and CompactRIO

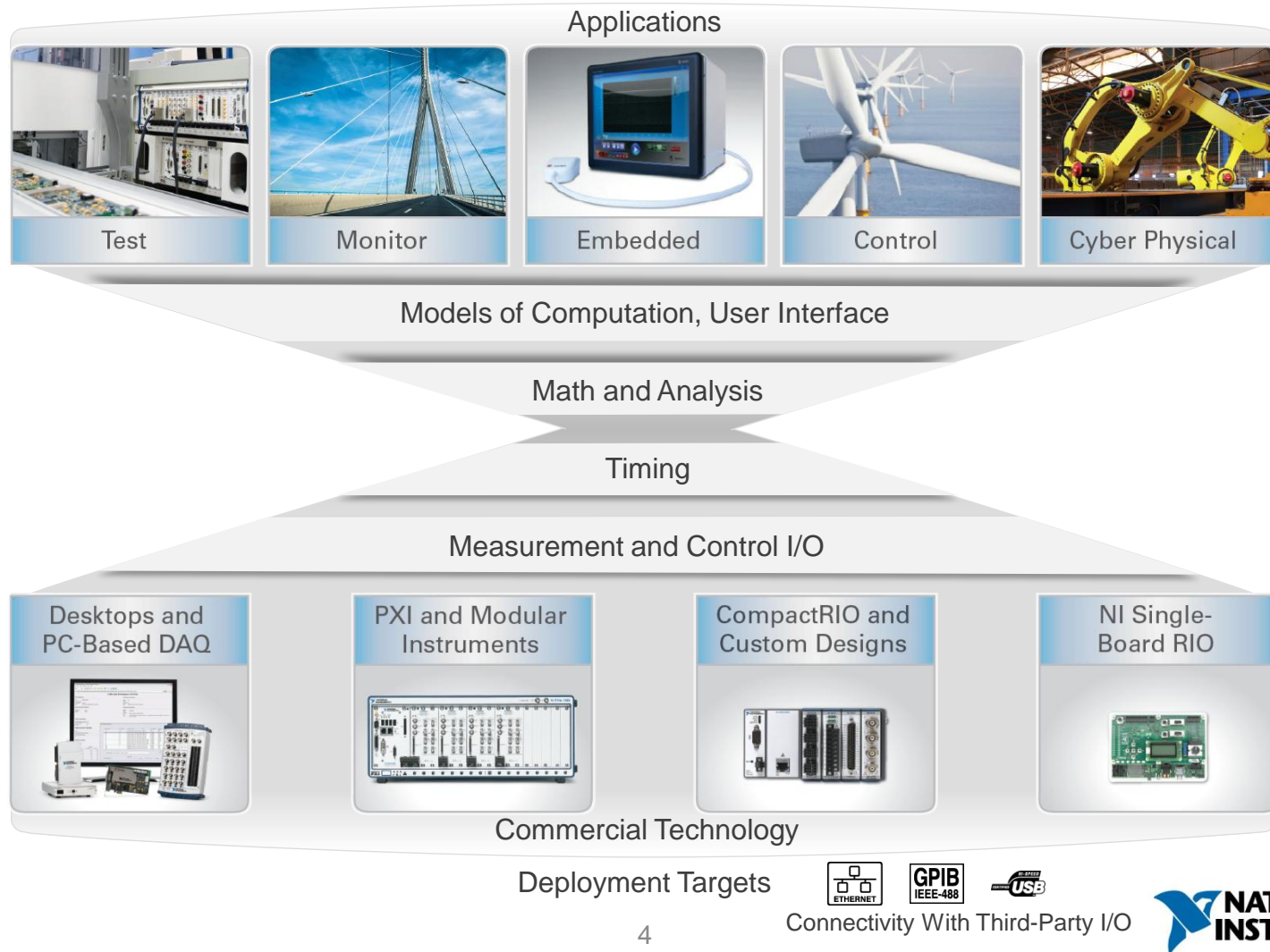


We all have a challenge to solve...



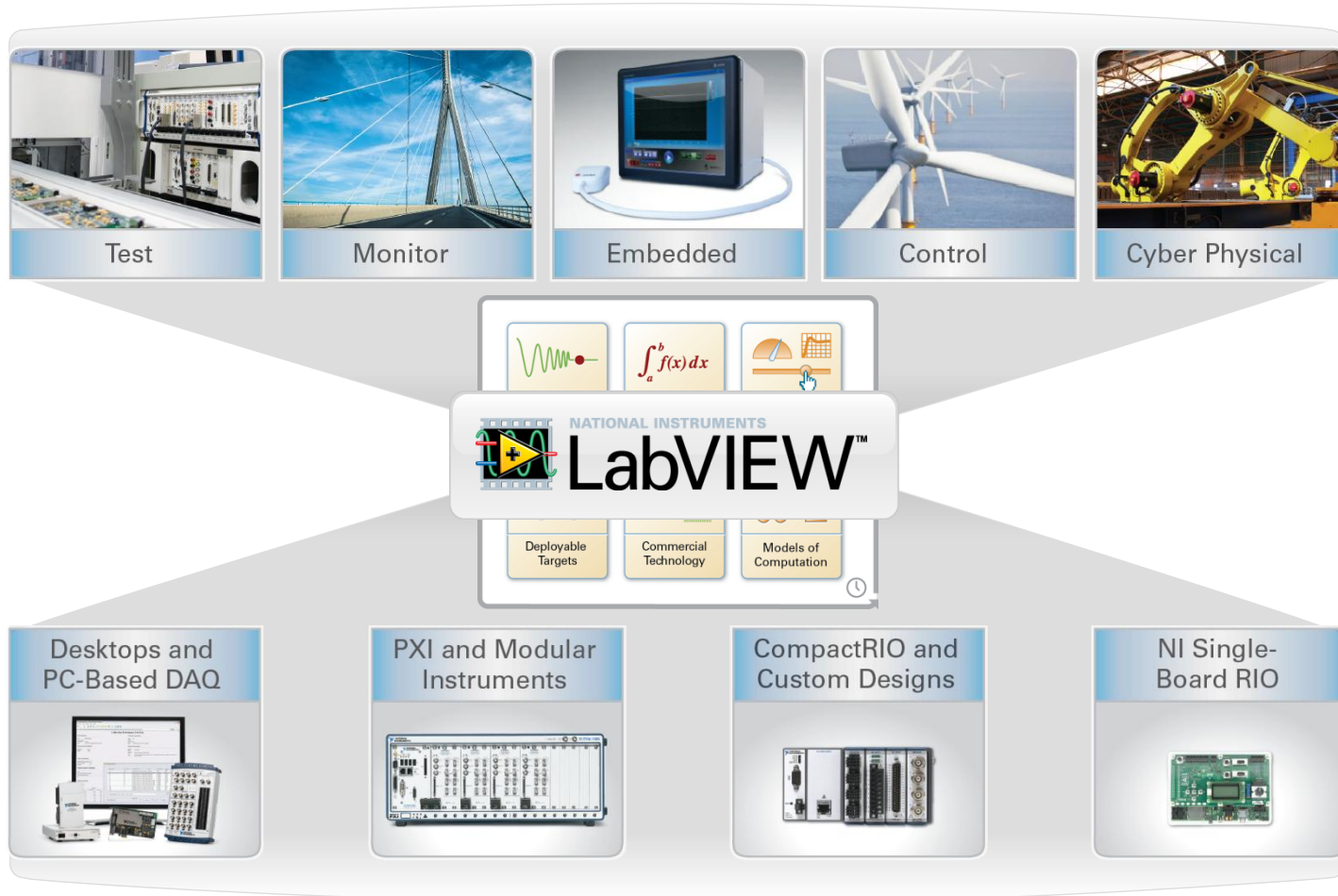
Graphical System Design

A platform-based approach for measurement and control



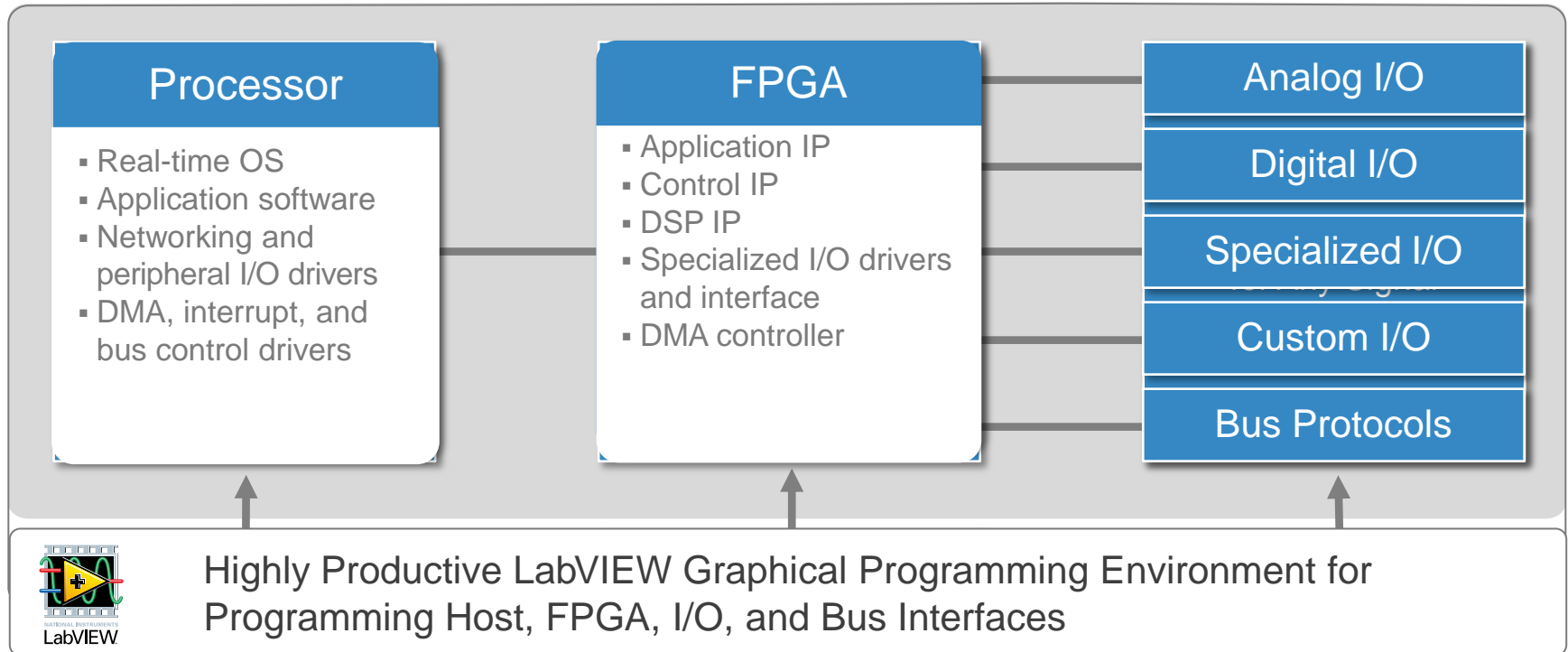
Graphical System Design

A platform-based approach for measurement and control



The LabVIEW RIO Architecture

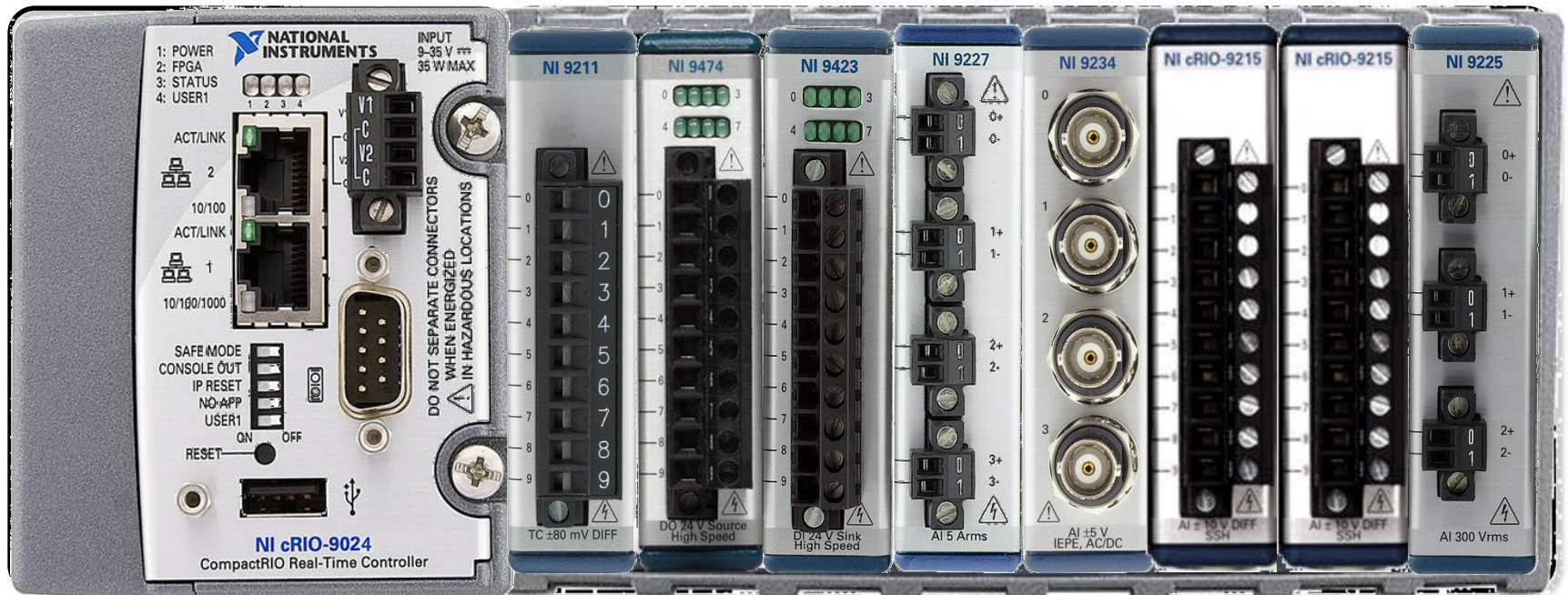
We call this the LabVIEW RIO Architecture.



The NI RIO System

REAL-TIME CONTROLLER

FPGA CHASSIS



MEASUREMENT MODULES

Operating System Characteristics

	<i>Loop Rate</i>	<i>Jitter</i>
General Purpose OS <ul style="list-style-type: none">• High-priority tasks can be preempted by lower-priority tasks• Extraneous background programs<ul style="list-style-type: none">- Screen savers, disk utilities, virus software, and so on• Peripheral Interrupts<ul style="list-style-type: none">- Mouse, keyboard, and so on	<i>10-100 Hz</i>	<i>Unbounded</i>
Real-Time OS <ul style="list-style-type: none">• Scheduler ensures high-priority tasks execute first• Direct control over all tasks• Stand-alone<ul style="list-style-type: none">• no mouse, keyboard, and so on	<i>Up to 50 kHz</i>	<i>Bounded</i>

Why is an Real-Time Operating System Useful?

- Strictly prioritizing tasks vs. design for fairness and user responsiveness
- Maximum reliability / uptime rather than on smooth multitasking
- Need more control over what the operating system spends its time executing

What is Real-Time?

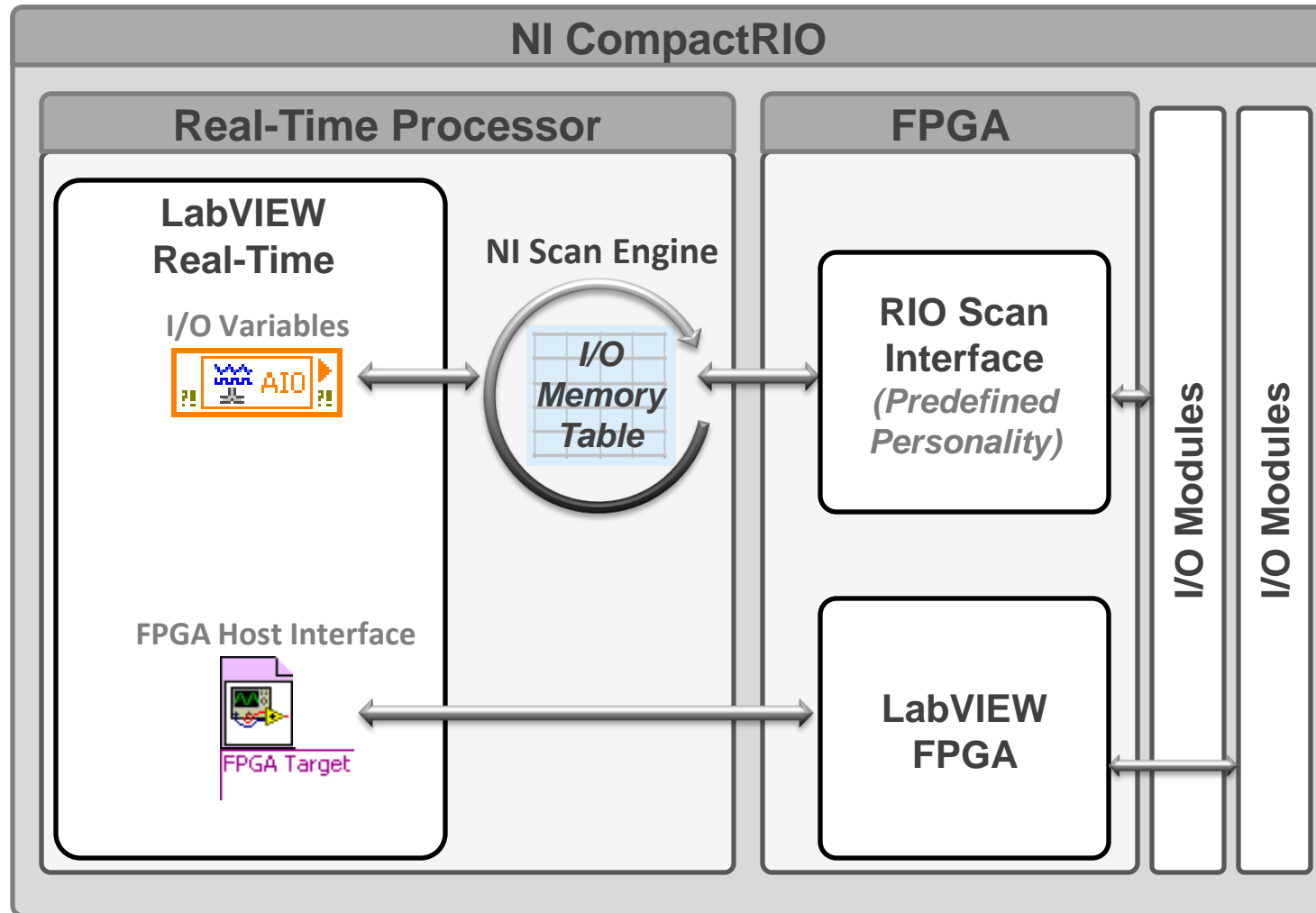
- Real-time **does not** always mean really fast
- Real-time means **absolute reliability**
- Real-time systems have timing constraints that must be met to avoid failure
- Determinism is the ability to complete a task within a fixed amount of time



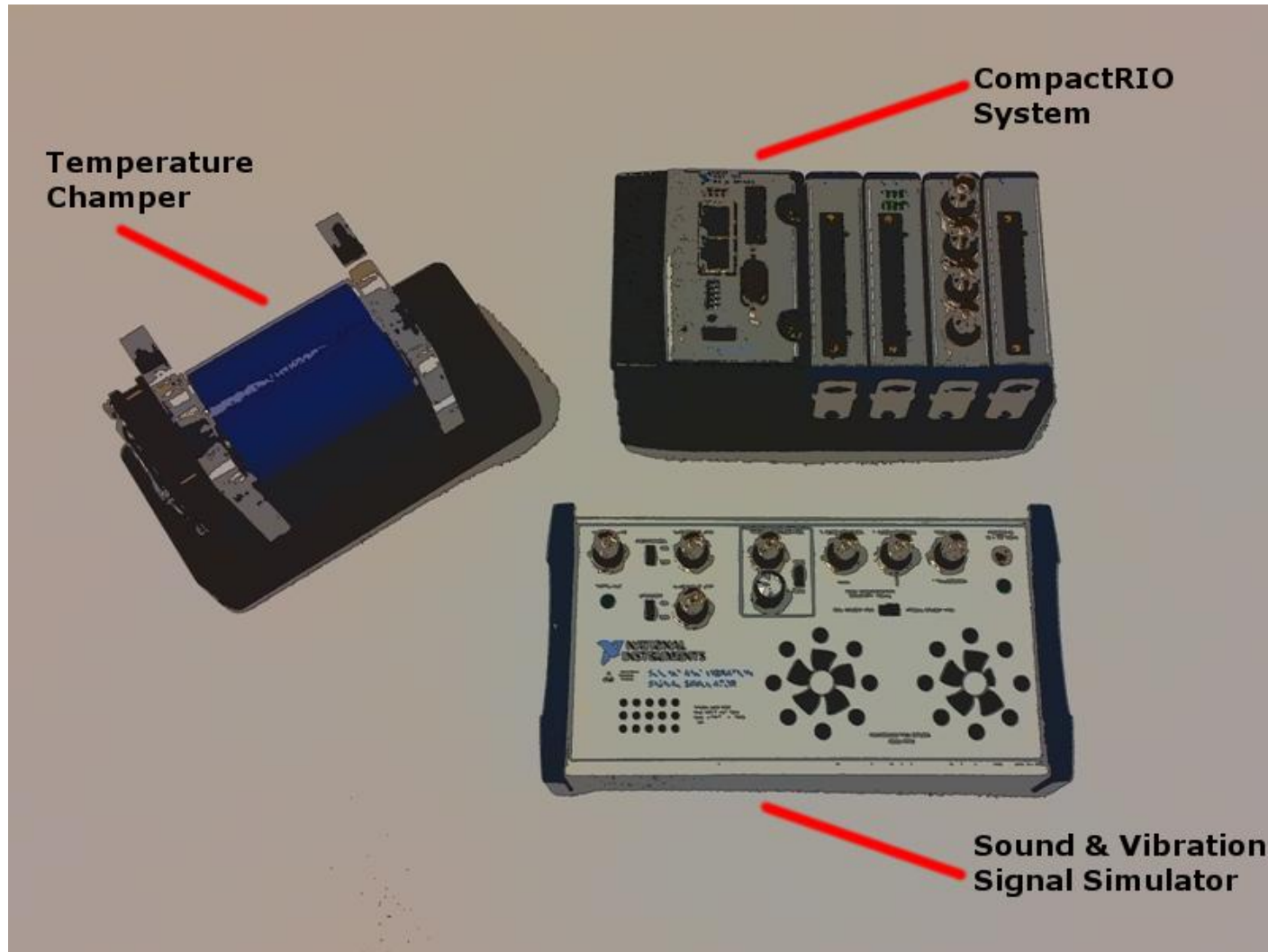
Why Are FPGAs Useful?

- ***True Parallelism*** – Provides parallel tasks and pipelining
- ***High Reliability*** – Designs become a custom circuit
- ***High Determinism*** – Runs algorithms at deterministic rates down to the order of nanoseconds
- ***Reconfigurable*** – Create new and alter existing task-specific personalities

LabVIEW Programming Model for CompactRIO



Today's System: The CompactRIO Hands On Kit



Hands-on Exercises:

- **[Exercise 1 - Temperature Control]**

Implement a closed-loop PID algorithm with conditional output to control the temperature within a temperature chamber (scan engine)

- **[Exercise 2 - Vibration Monitoring]**

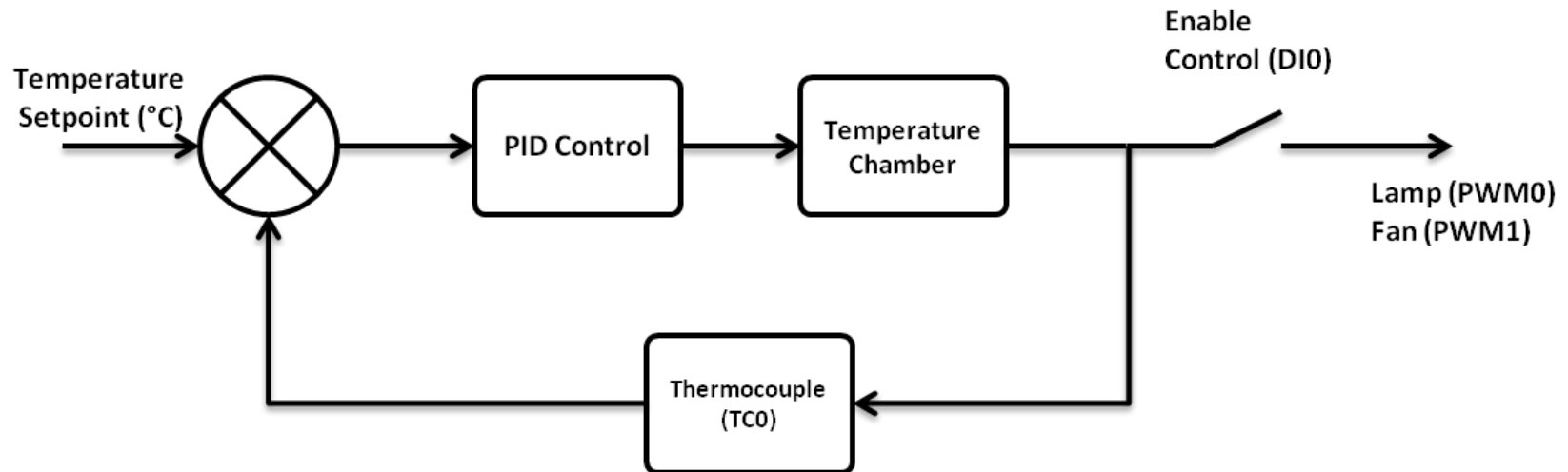
Use the FPGA portion of the CompactRIO system to implement a speed limit control algorithm for an unbalanced fan by monitoring signals from an accelerometer and a tachometer.

Develop RT-code that communicates with the FPGA.

- **[Exercise 3 - HMI and Communications]**

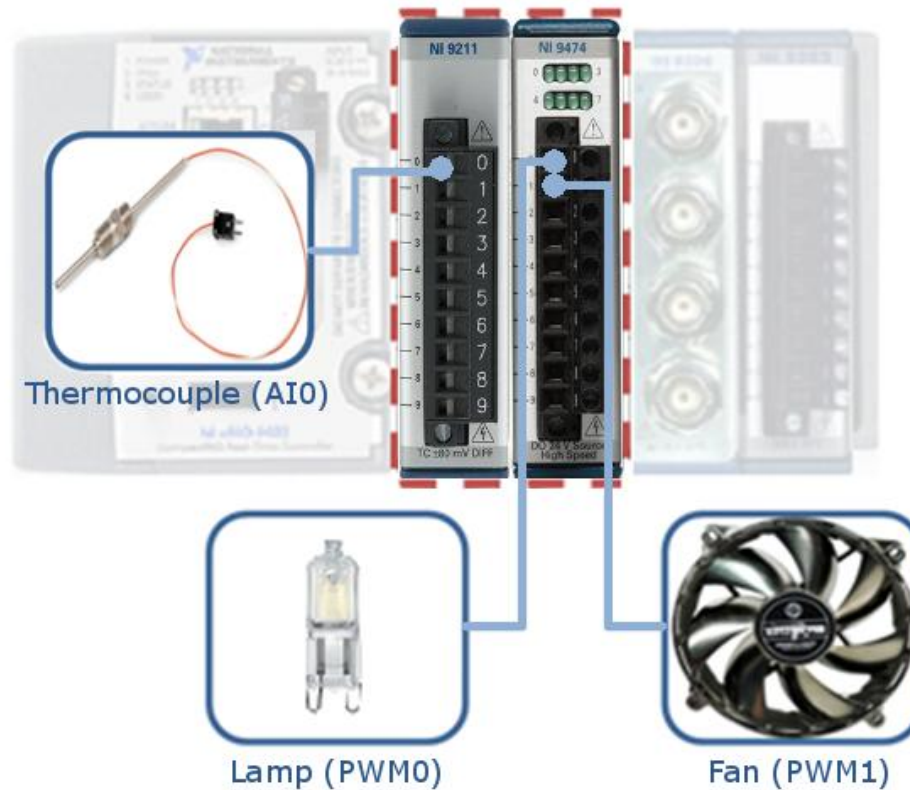
Create a remote HMI to visualize the data of the vibration monitoring system.

Temperature Control with Conditional Output of a Temperature Chamber



Exercise 1 - Temperature Control

Exercise 1 - Temperature Control

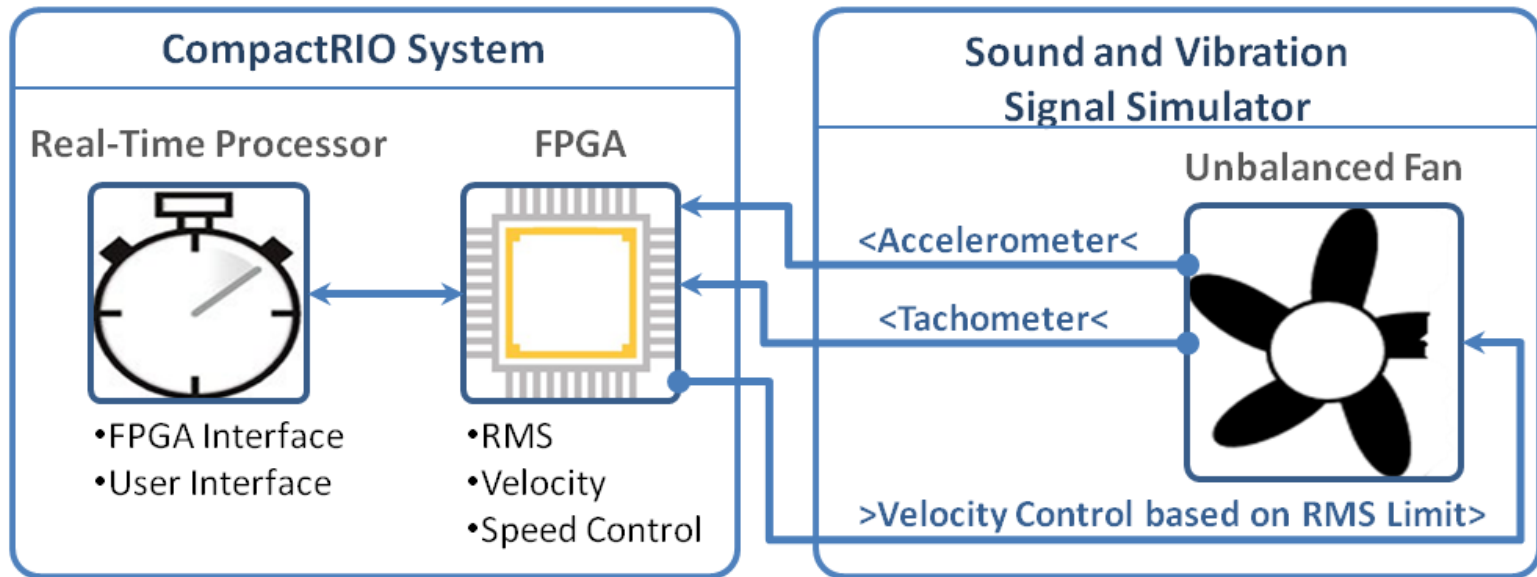


Inputs:

Slot 1- NI 9211-*Thermocouple Input*
Analog Input 0 (AI0)→ Thermocouple

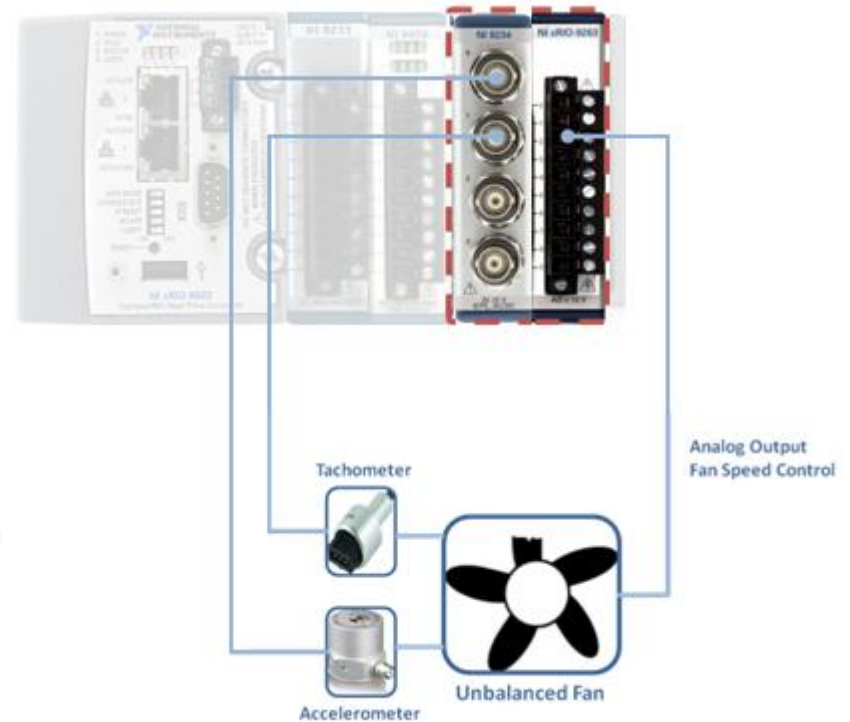
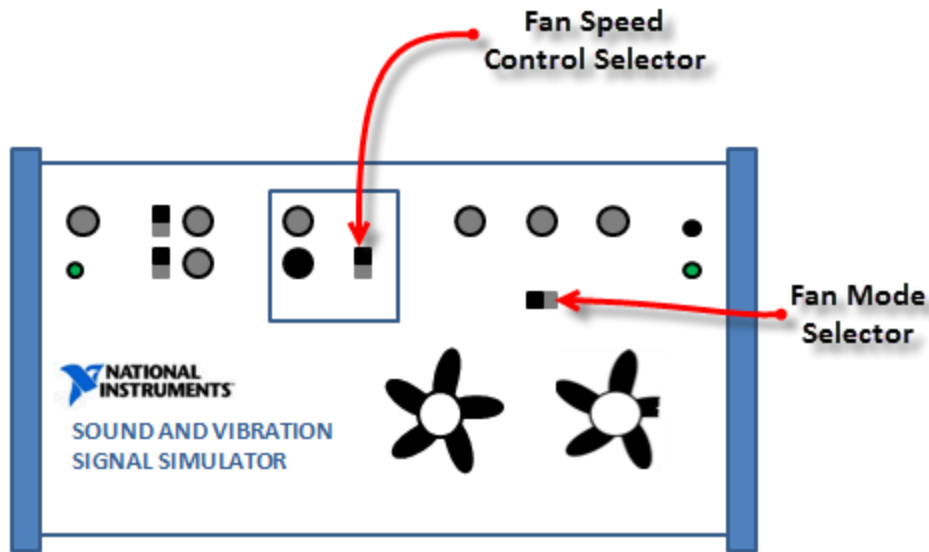
Outputs:

Slot 2- NI 9474-*Digital Output (PWM)*
Digital Output 0 (PWM0)→Lamp
Digital Output 1 (PWM1)→Fan



Exercise 2 - Vibration Monitoring

Exercise 2 - Vibration Monitoring



Inputs:

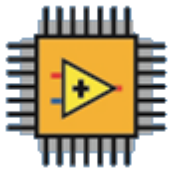
Slot 3: NI 9234-Dynamic Signal Acquisition Module

AI0 → Accelerometer
AI1 → Tachometer
Sampling Period: 51.2kS/s
Resolution: 24-bit

Outputs:

Slot 4: NI 9263-Analog Output Module

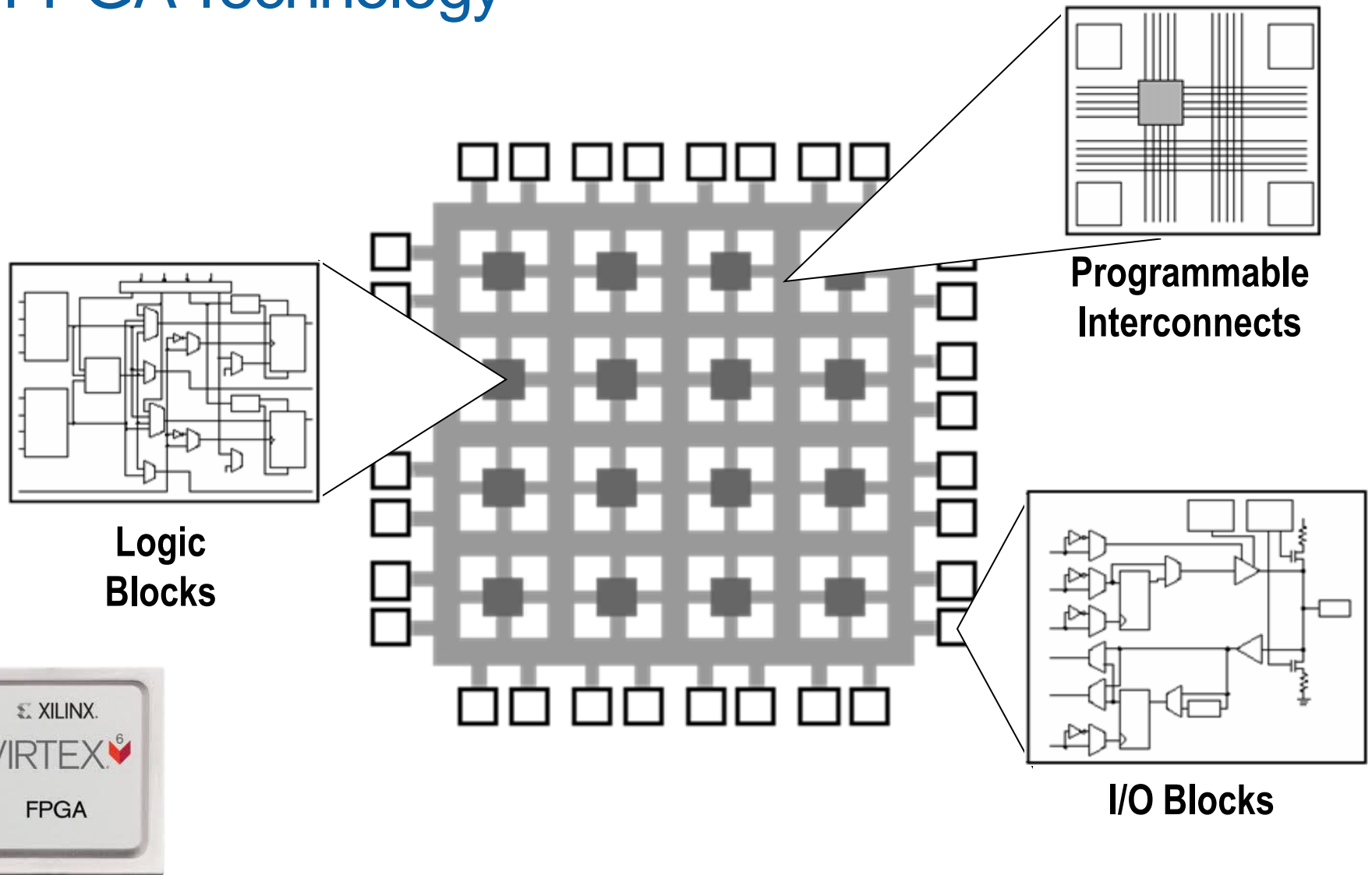
AO0 → Fan Speed Control
Sampling Period: 100kS/s
Resolution: 16-bit



Hands-on Embedded Systems 2014

Introduction to LabVIEW FPGA

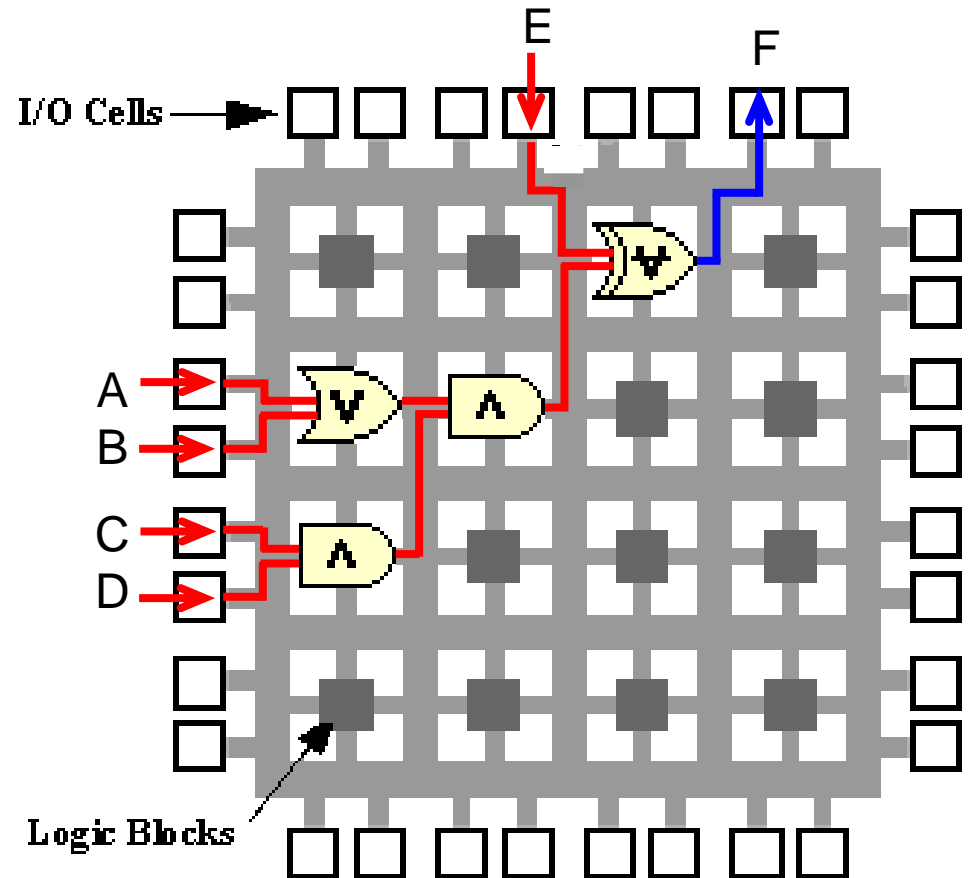
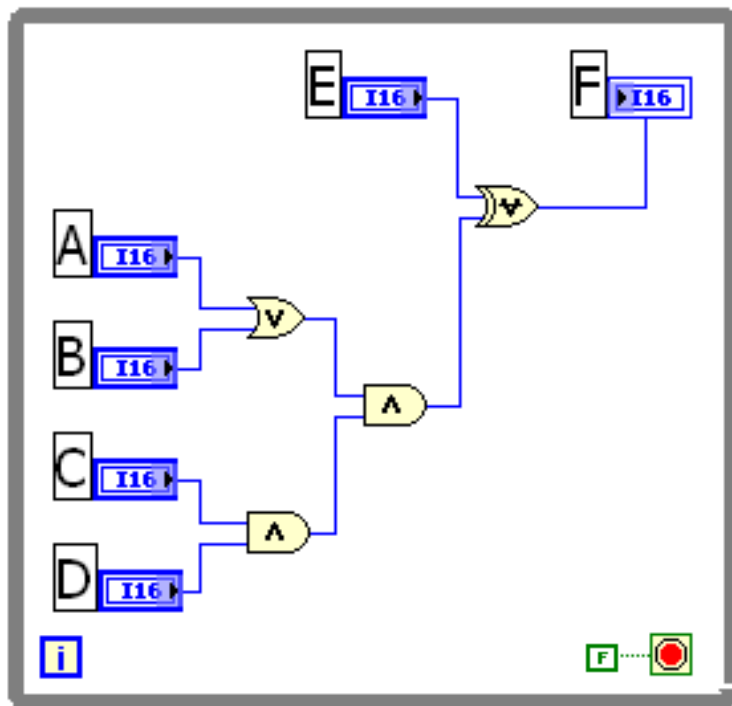
FPGA Technology



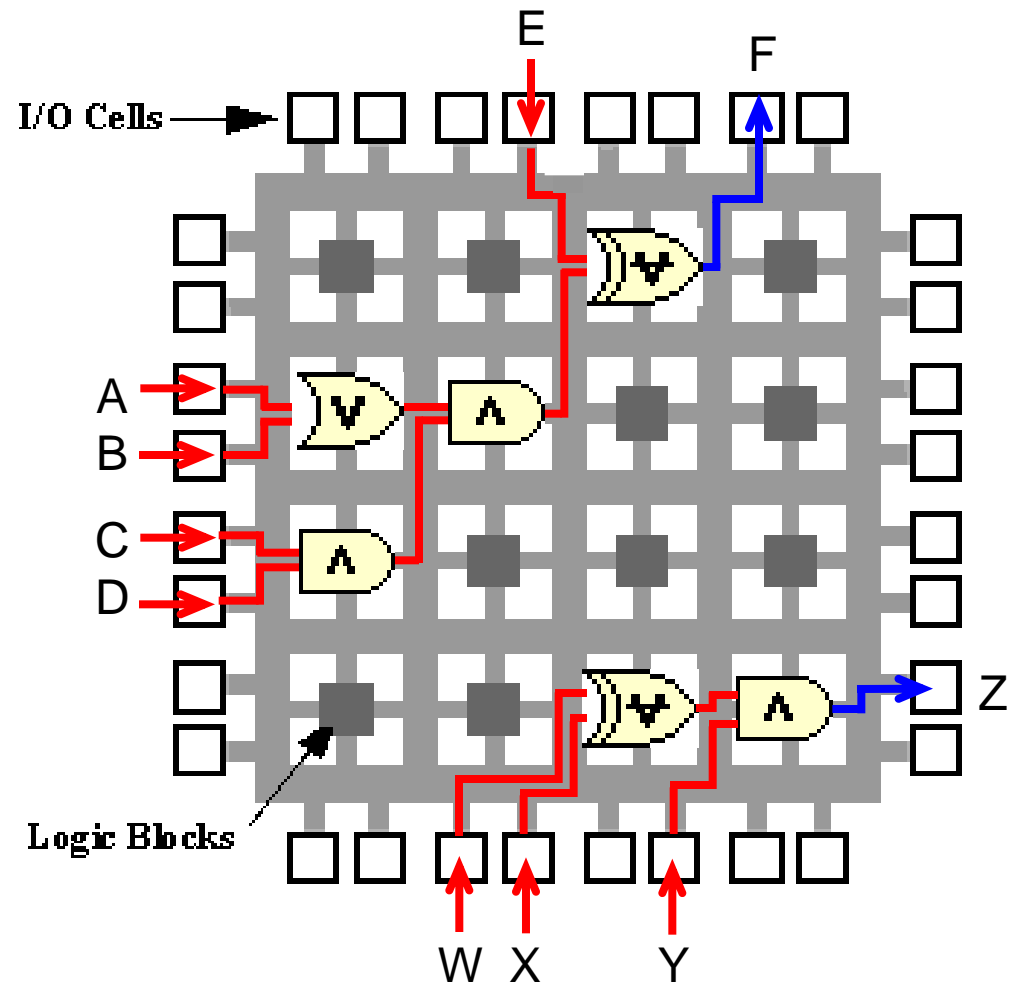
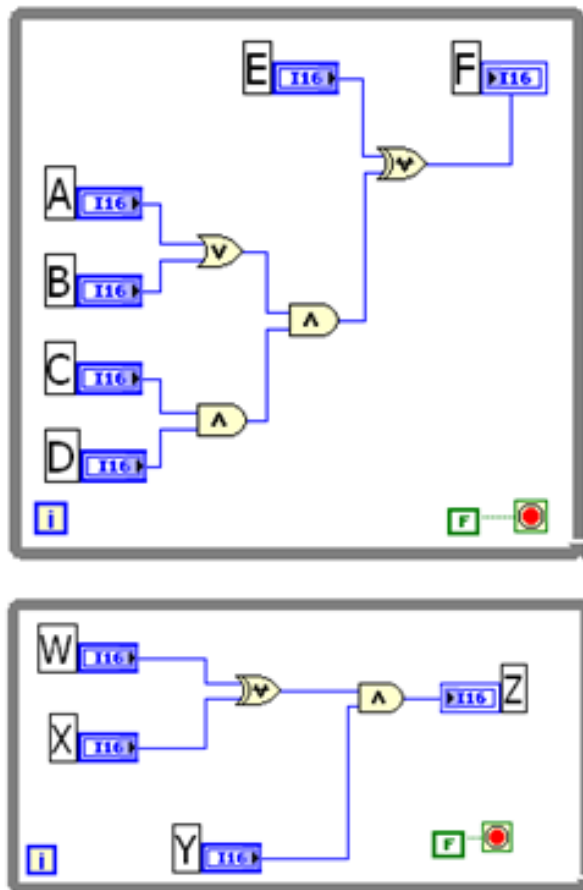
FPGAs are Dataflow Systems

Implementing Logic on FPGA: $F = \{(A+B)CD\} \oplus E$

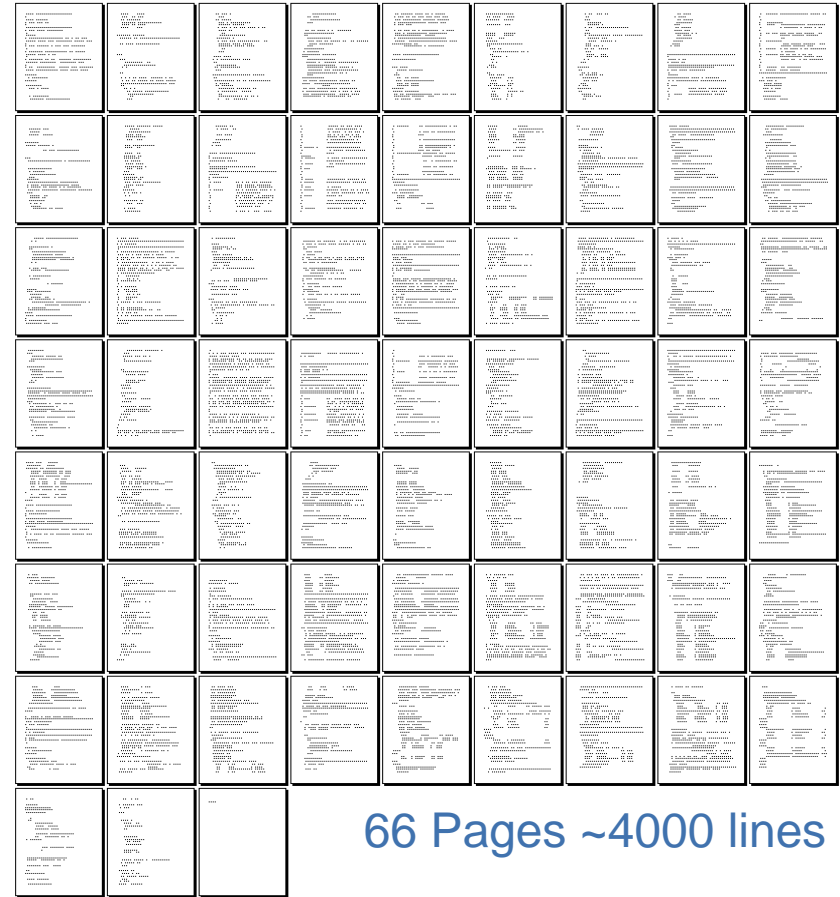
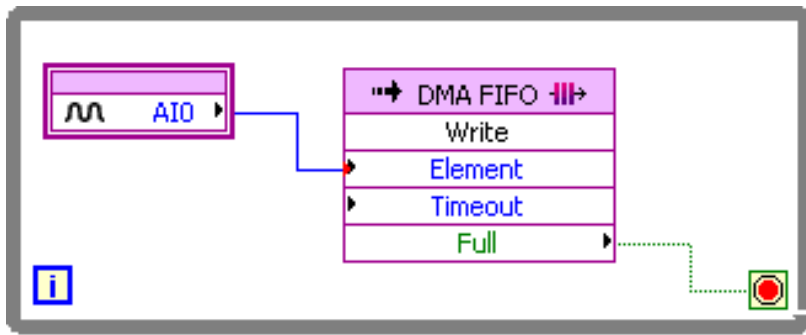
LabVIEW FPGA Code



FPGAs are Parallel Dataflow Systems



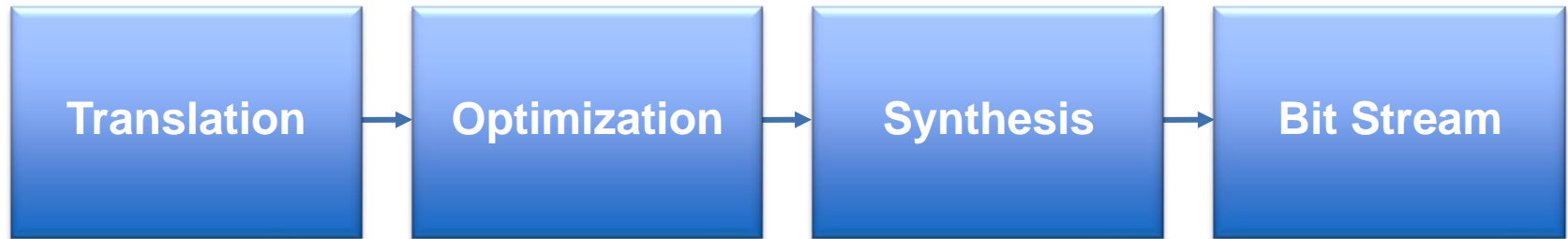
Middleware and I/O Drivers Included



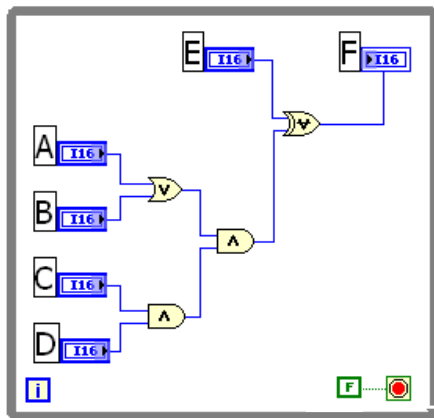
66 Pages ~4000 lines

LabVIEW FPGA vs. VHDL

Compilation Process



LabVIEW FPGA Code



Compile VHDL through Xilinx

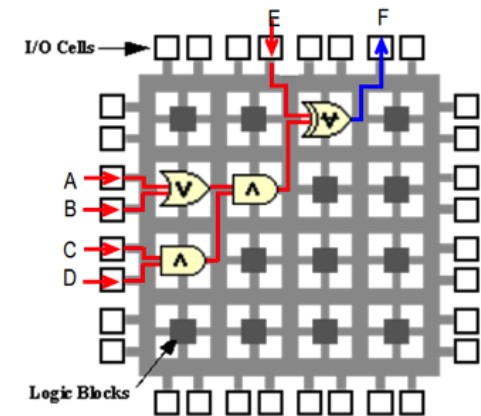
```
-- end if;
end process synchronizationFFs;

-- Then we keep track of what the digital input was on the previous
-- clock cycle by inserting another flip flop
PreviousDigitalInputFF:
process( aReset, Clk )
begin
    if aReset then
        cPrevDigitalInput <= false;
    elsif rising_edge(Clk) then
        cPrevDigitalInput <= cDigitalInput;
    end if;
end process PreviousDigitalInputFF;

-- Then we have a little combinatorial logic to detect a rising edge
cRisingEdgeDetected <= cDigitalInput and not cPrevDigitalInput;

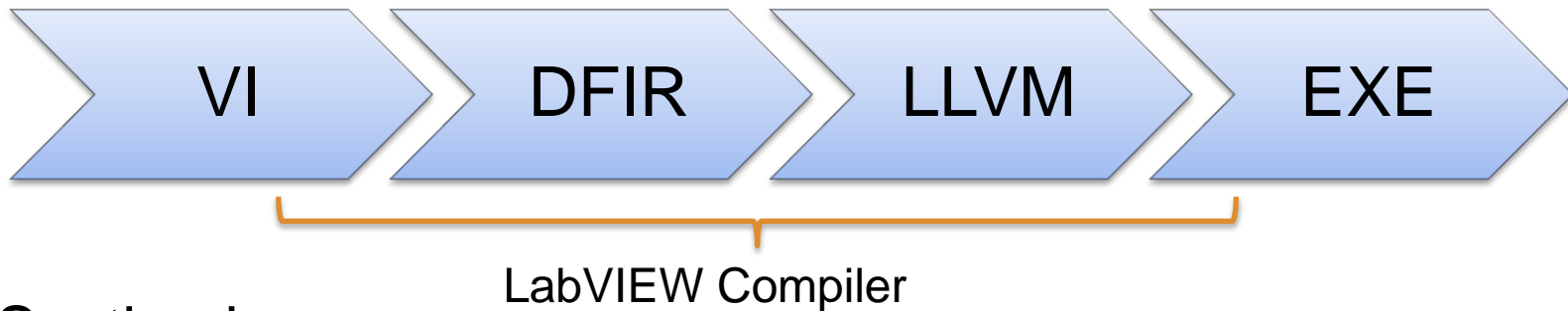
-- And finally we have a register that increments when that rising
-- edge is detected.
CounterRegister:
process( aReset, Clk )
```

FPGA Logic Implementation

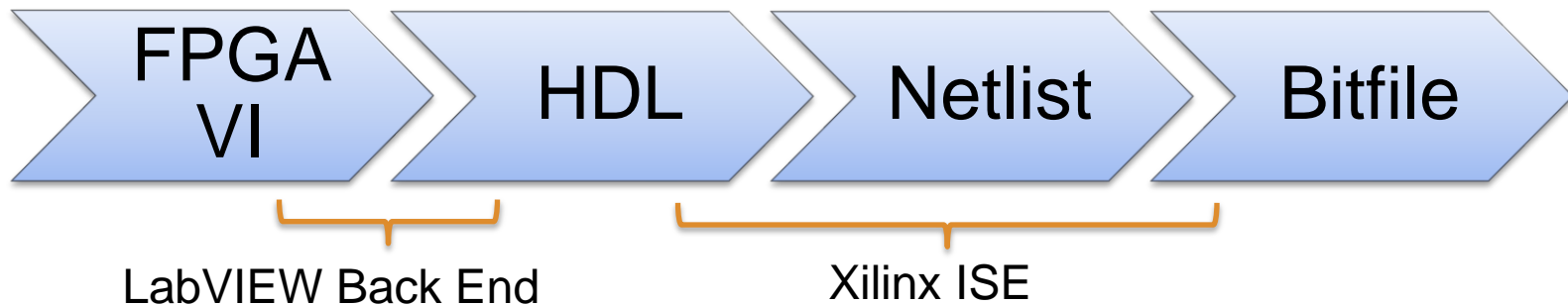


Compilation Process

Compilation

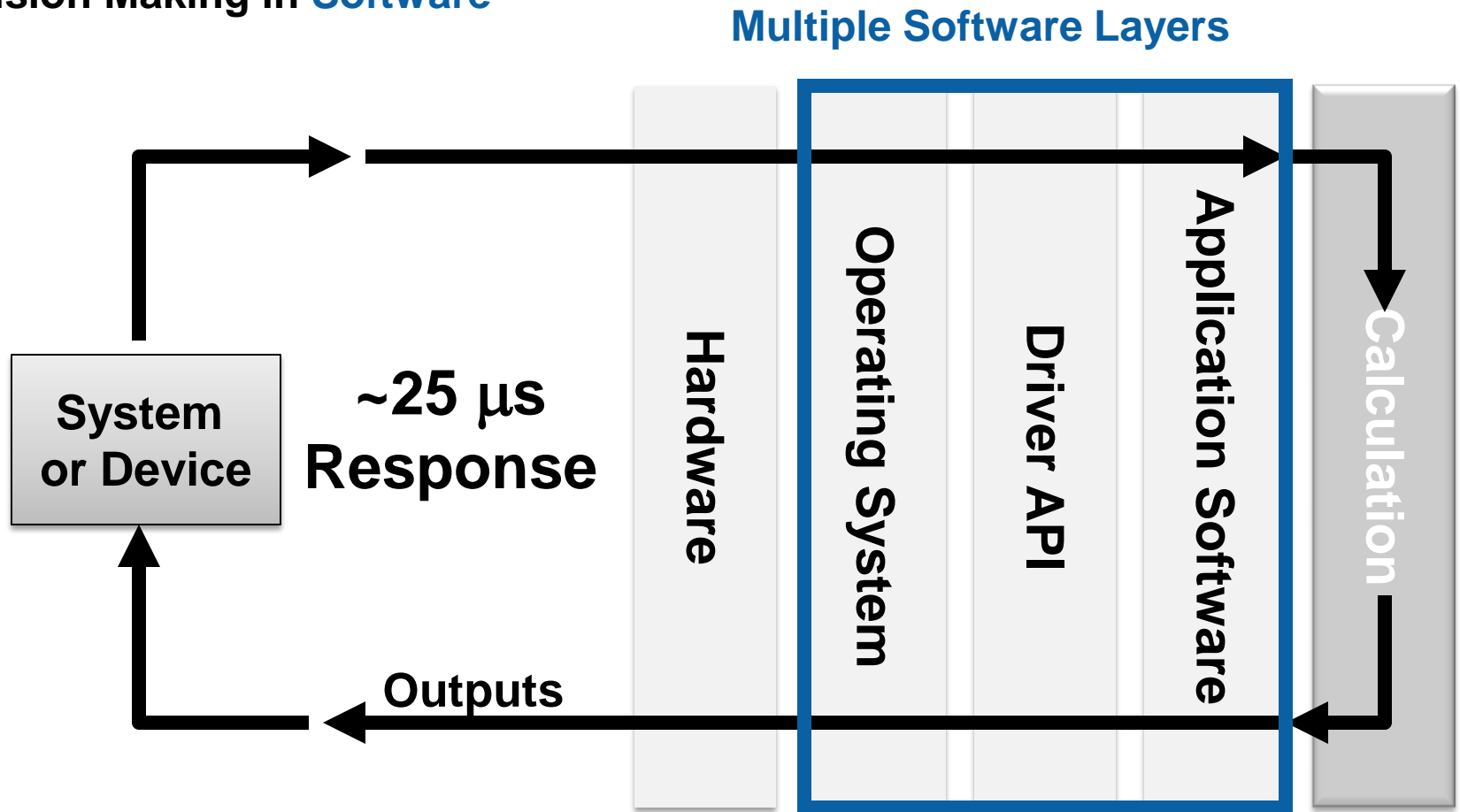


Synthesis



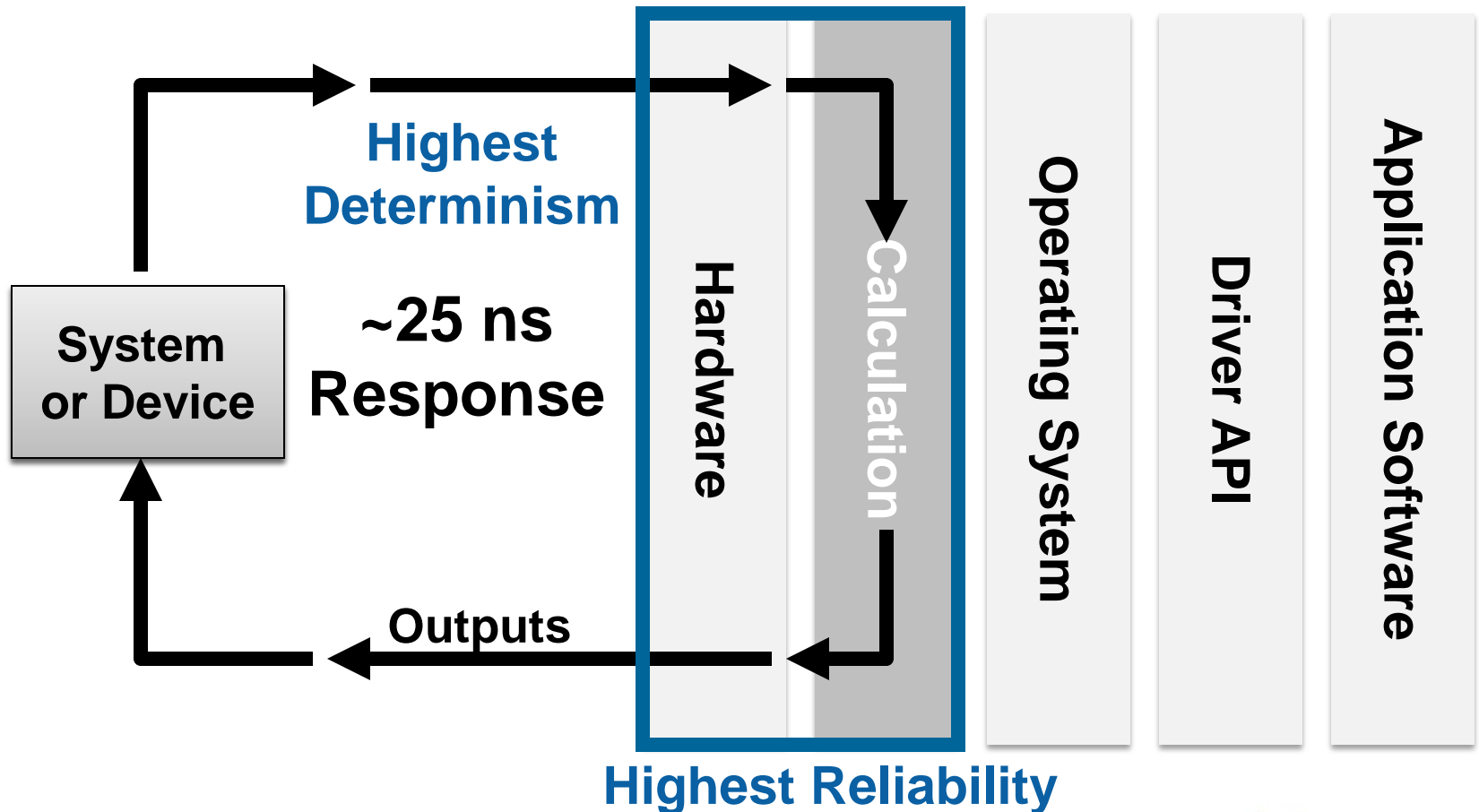
High Reliability and Determinism

Decision Making in Software



High Reliability and Determinism

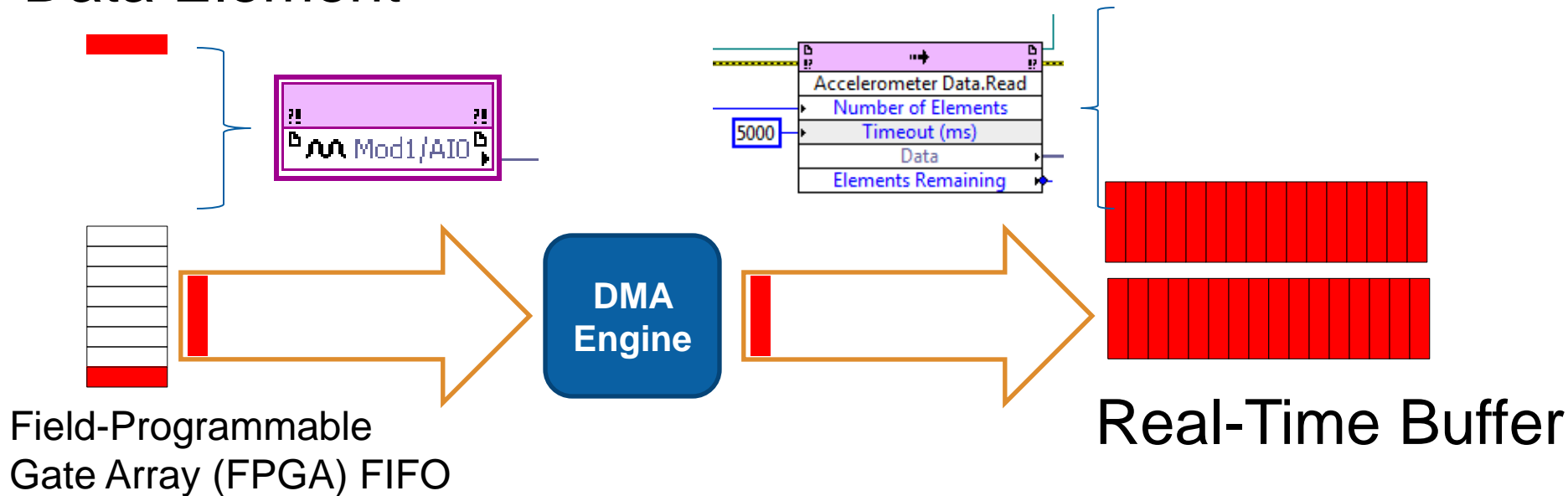
Decision Making in Hardware

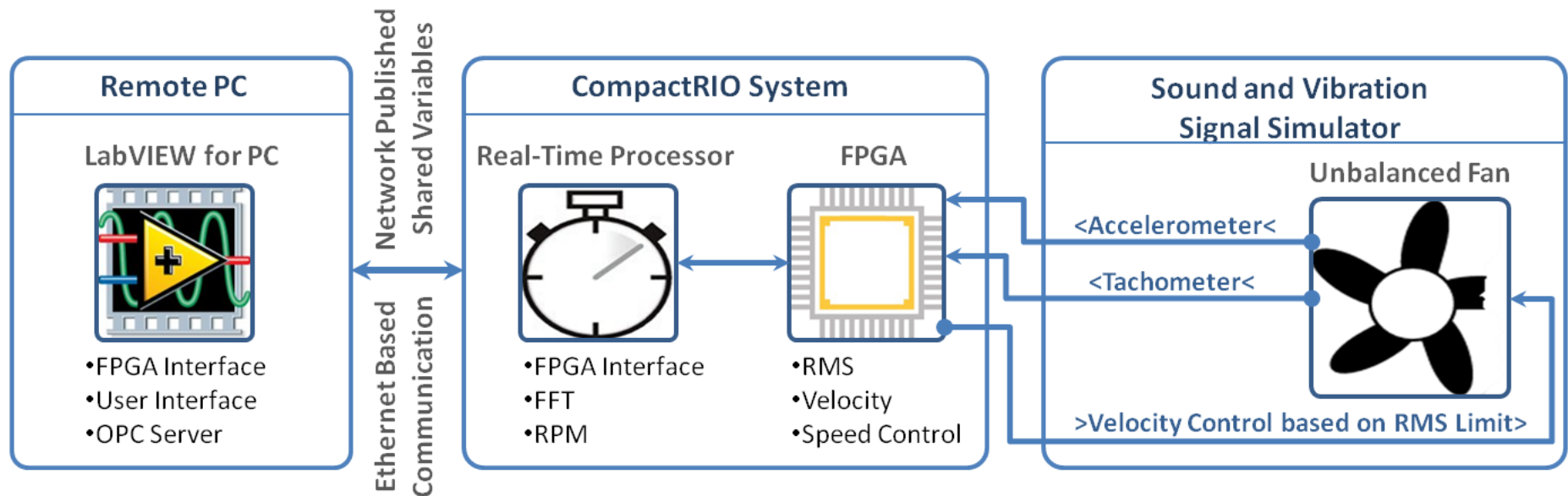


Transferring Data to the RT Microprocessor



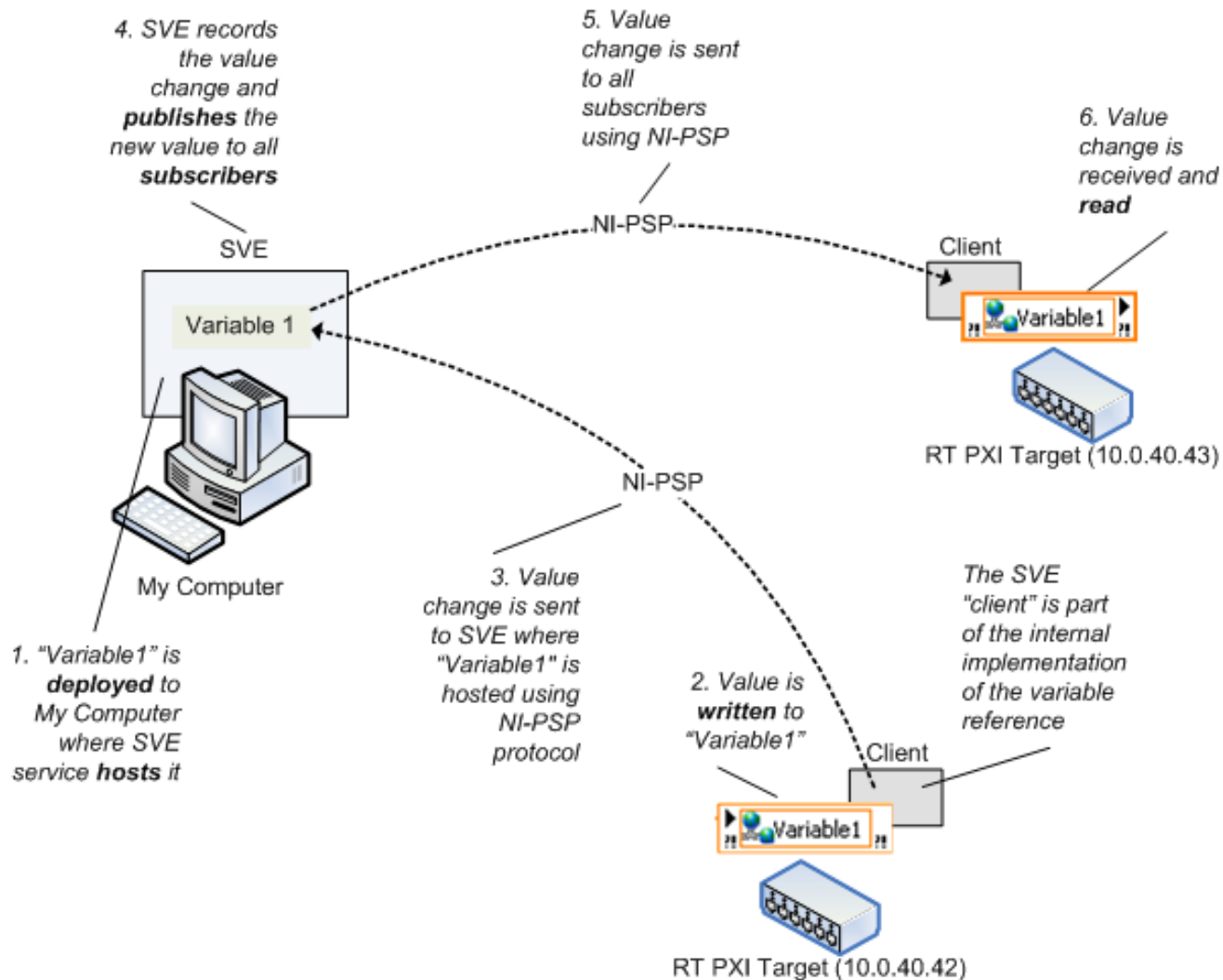
Data Element



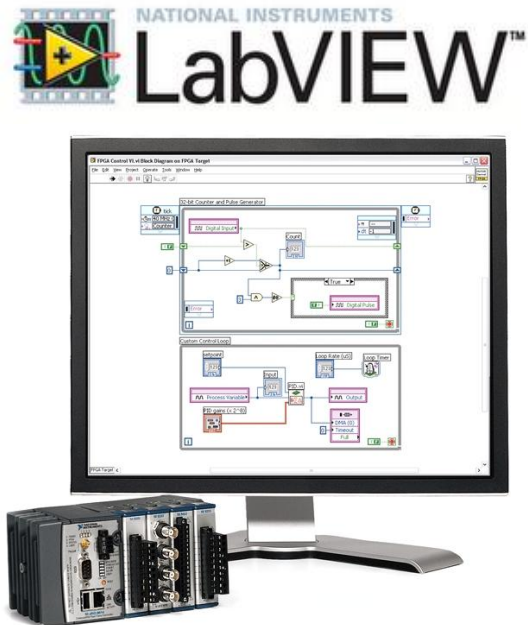


Exercise 3 - HMI and Communications

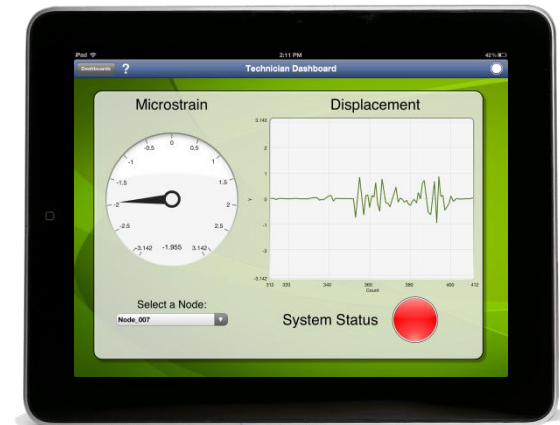
How a Network-Published Shared Variable Works



Monitor and control your application from your tablet or smart phone: Data Dashboard.



Data Dashboard for LabVIEW



Available on the
App Store

Control and visualize data from LabVIEW systems on an iPad

LabVIEW RIO Hardware

CompactRIO and NI Single-Board RIO



Value



Ultra Rugged



Performance

PXI, PC RIO (R Series, NI FlexRIO)



High Performance

Expansion I/O



MXI-Express RIO



Ethernet RIO



EtherCAT RIO



Wireless

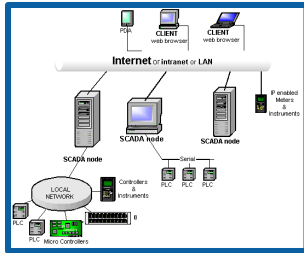
Connect to Any Sensor on Any Bus

100+ Industrial I/O Modules

- ✓ Accelerometer
- ✓ Strain gage
- ✓ Resistance
- ✓ Load cells
- ✓ Digital I/O and protocols
- ✓ Microphone
- ✓ Bus communications
- ✓ Thermocouples
- ✓ 4 to 20 mA
- ✓ Storage media
- ✓ RTD
- ✓ Engine control
- ✓ Industrial vision
- ✓ Motion control



Third-Party Communication



Proprietary Protocols
PROFIBUS
EtherCAT
CAN & CANopen

Connectivity

“Open” Protocols
Modbus
Serial
OPC Classic
OPC UA

Deployment Targets



NATIONAL INSTRUMENTS

LabVIEW™

Analysis
Visualization
Data-logging



Use LabVIEW and RIO when

It is important to...



Solve complex applications, especially where ruggedness, high-performance, reliability, and quality measurements are important.

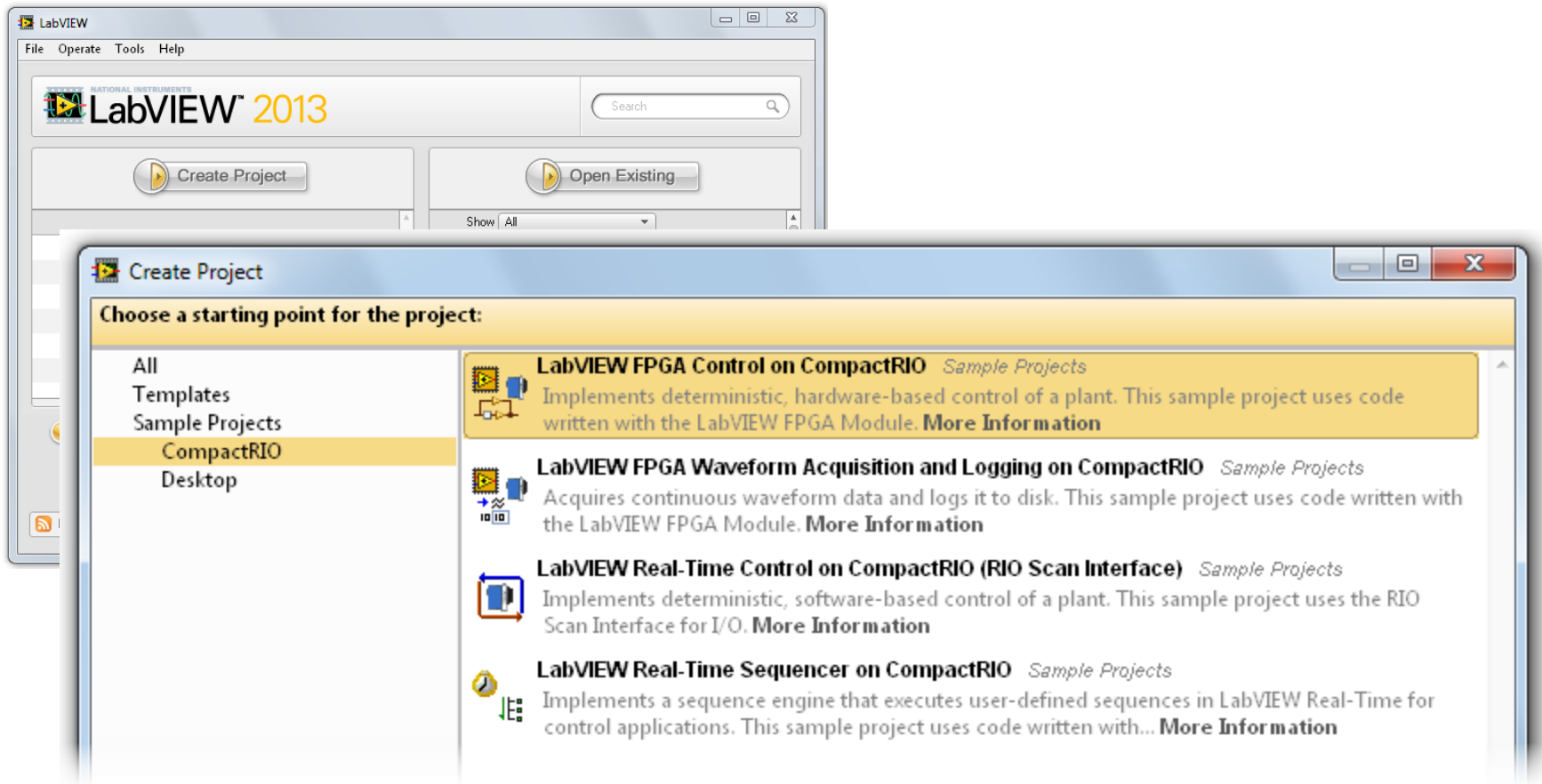


Avoid the negative impacts of design changes and updates with flexible hardware.



Get to market faster with high-level system design software, existing IP, and integrated hardware and software.

LabVIEW 2013 RIO Sample Projects



CompactRIO Developer's Guide

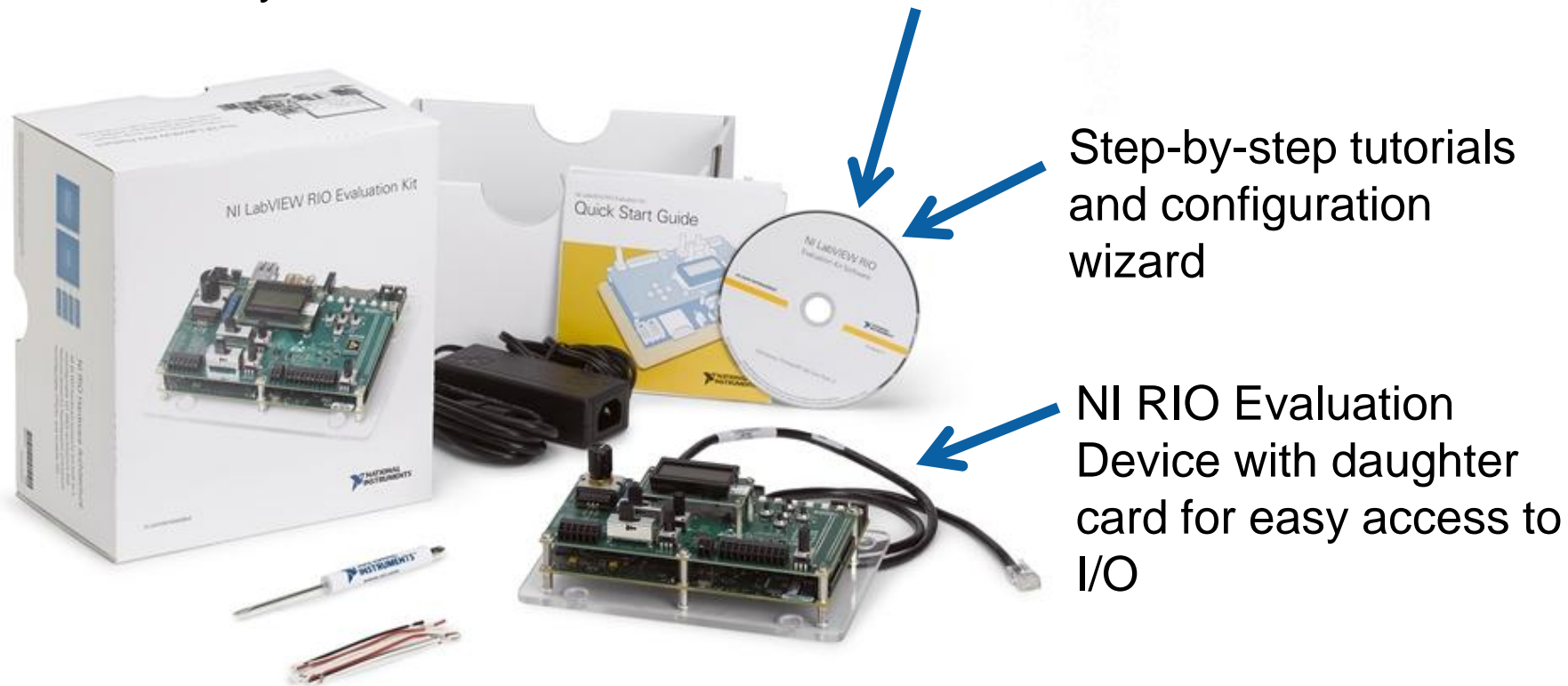
NI LabVIEW for CompactRIO Developer's Guide

Recommended LabVIEW Architectures and Development Practices
for Control and Monitoring Applications

<http://www.ni.com/compactriodevguide/>

NI LabVIEW RIO Evaluation Kit

90-day LabVIEW, LabVIEW FPGA & LabVIEW Real-Time evaluation

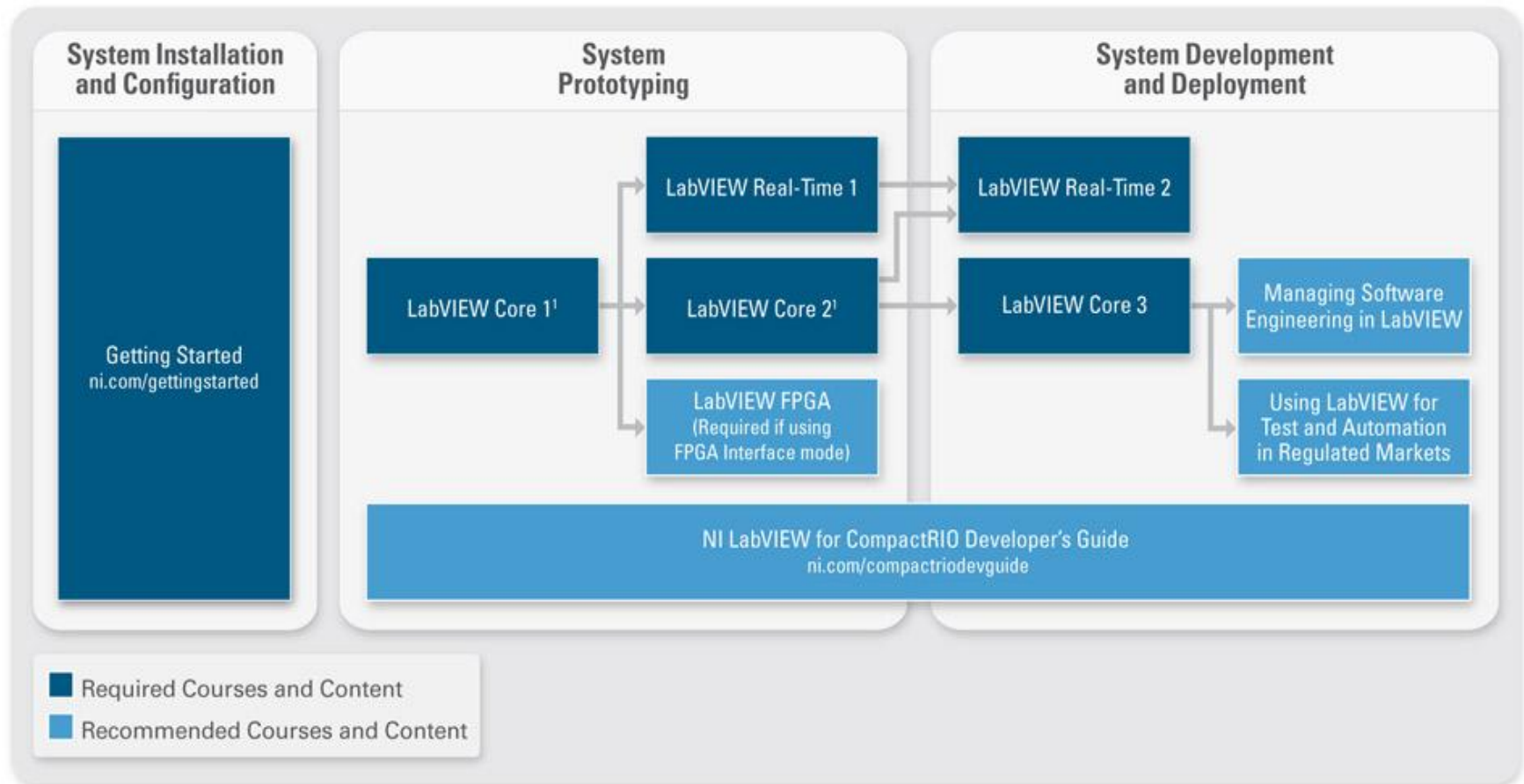


Step-by-step tutorials
and configuration
wizard

NI RIO Evaluation
Device with daughter
card for easy access to
I/O

Order at ni.com/rioeval
Online Community at ni.com/rioeval/nextstep

Training & Certification Path for Embedded



¹ Since LabVIEW Core 1 and 2 fit into one week, you may choose to take both LabVIEW Core classes prior to LabVIEW FPGA and LabVIEW Real-Time 1.

ni.com/self-paced-training

Alliance Partner Network

- Programming
- Project management
- Component design
- Integration
- Consulting services
- Product ecosystem
- Specialty I/O

700+

Over 700 Companies



Twenty Years
of Partnerships



Expert Knowledge



Partner Certification



Worldwide



Partner Search



Tools should not limit
Innovation and Discovery

