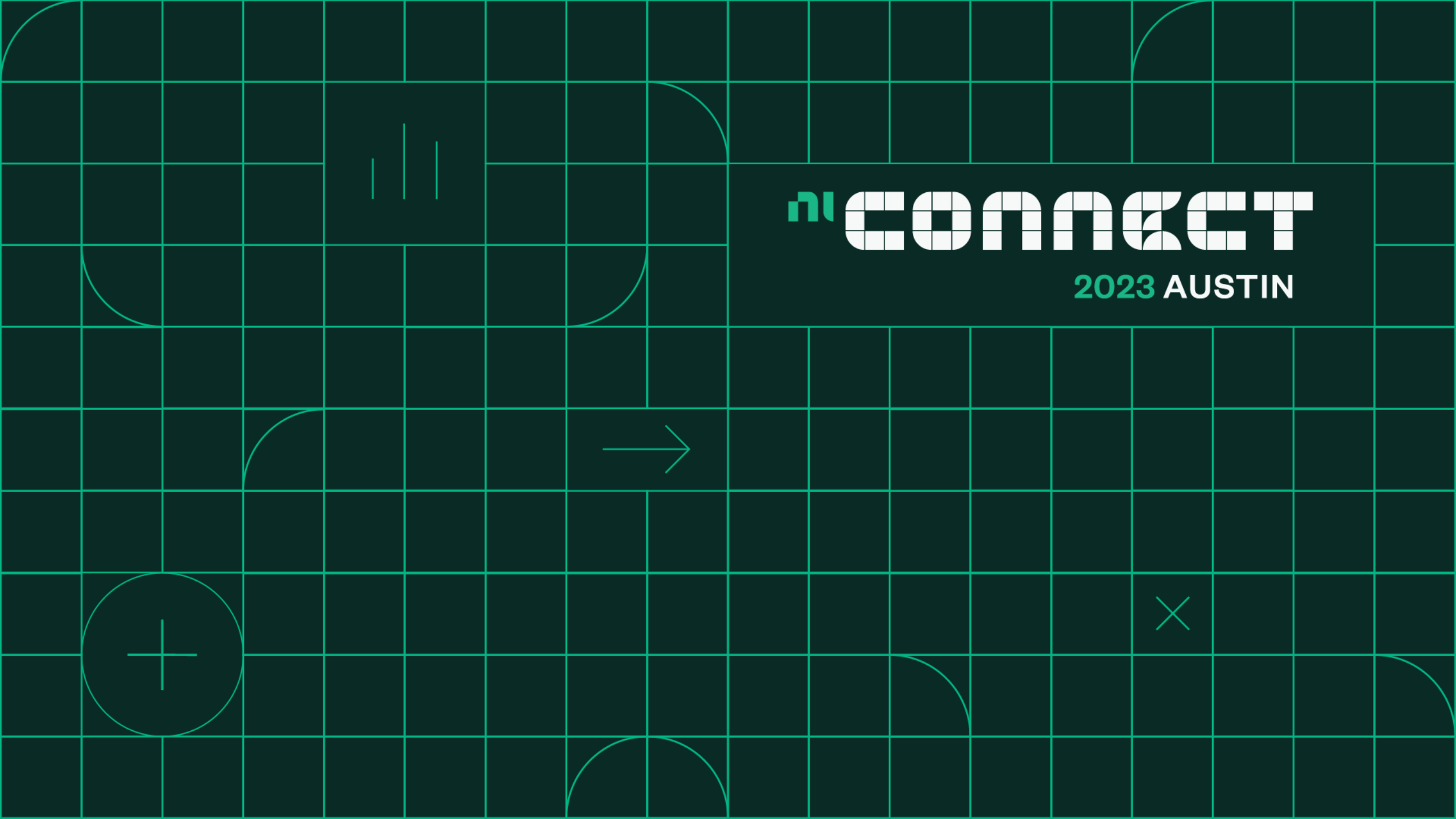

CONNECT
2023 AUSTIN

Introductions

Introduction: James Thornton

Senior Product Manager

Platform Product Management

Experience:

- National Instruments (1 year)
  - Products Managed
    - ATE CC, PXI Systems, GPIB, Serial, C-series controllers and Chassis
- Engineering Automation (6 years)
  - Roles
    - R&D, Technical Sales, Product Management

Introduction: Ryan Vallieres

Bloomy Controls, Inc (~10 years)

Senior Embedded Software Engineer

Embedded SW Engineering Supervisor

Experience:

- Large-scale distributed applications
  - 1000s of VIs across multiple targets
- Time-critical applications
  - Nano-second synchronization
- Mixed-platform solutions
  - LinuxRT, FPGA, Windows
  - Embedded programmable controllers

What are we doing here today?

Timing & Sync Terminology

Simple Communication with Hardware

Synchronization Types

Signal-Based Synchronization

- No external time reference
  - May still have a reference clock
- System is all physically connected
- Offers highest precision of synchronization
- Limited to cable distances of up to 200m
  - Length-matched cables are important!

Protocol-Based Synchronization

- Uses an external time reference
- IEEE-1588, GPS, IRIG-B (for example)
- Useful for systems with high node counts
  - Non-homogenous systems benefit as well
- Nodes are connected through a network
  - Node distance is protocol-dependent

Timing & Sync Terminology

Signal-Based Synchronization

- Clocks
- Triggers
- Drift
- Skew
- Trigger Skew Correction
- Jitter

Clocks

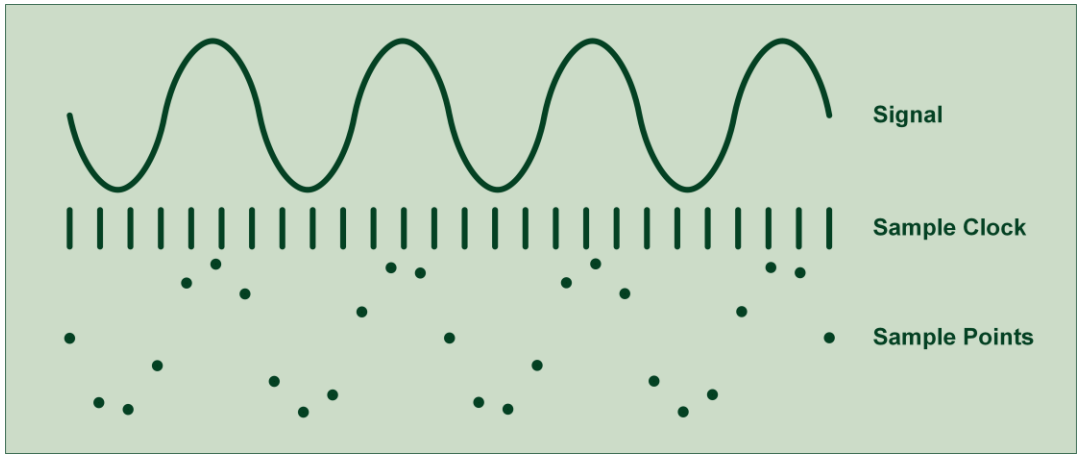
Periodic digital signal. Provides the “timing” for an acquisition event.

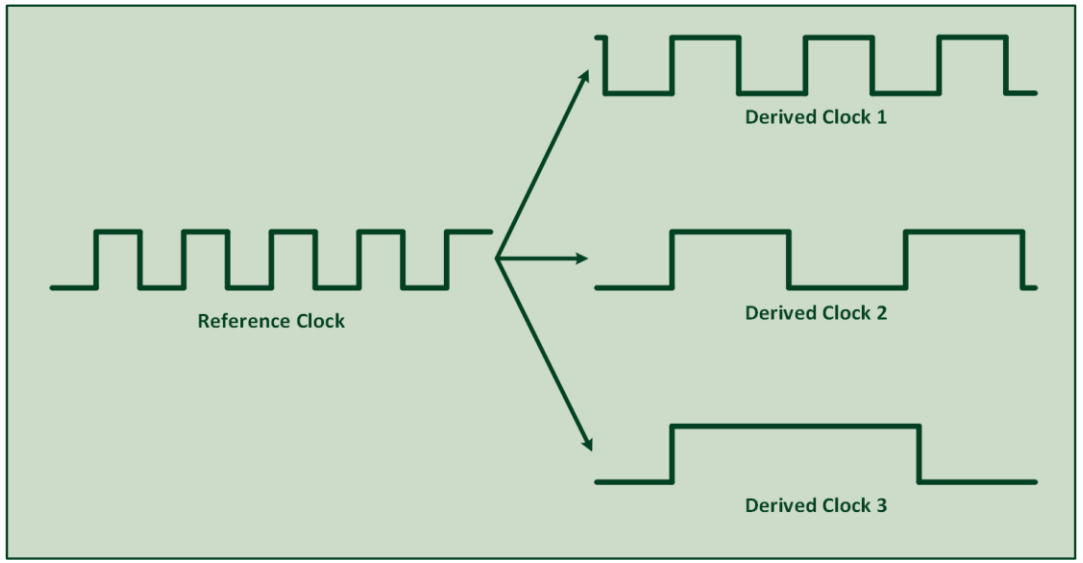
Clock Types: Reference Clock, Sample Clock

Clock Sources: Local*, Remote*, Onboard

*These can be relative to the acquisition target!

This is our “conversation” with the hardware. We’re exchanging information with the computer.



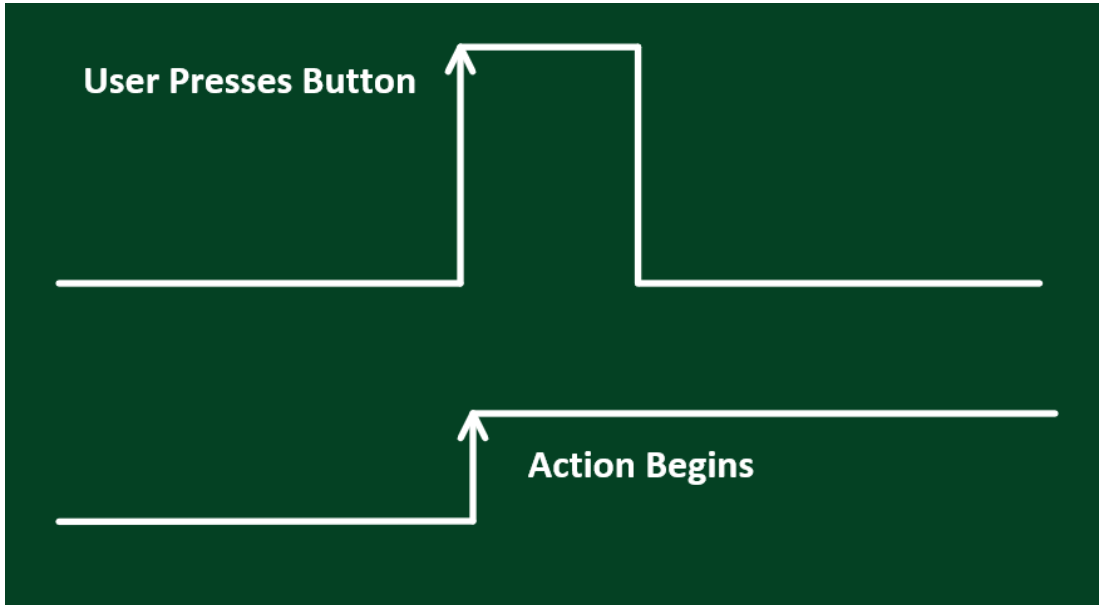


Triggers

A source for an event in acquisition. Can be either software or hardware.

Event Types: Start, Stop, Pause, etc…

Trigger Sources: Local, Remote, Onboard, Future Time Event



Akin to a "greeting". We're engaging our hardware in an activity.

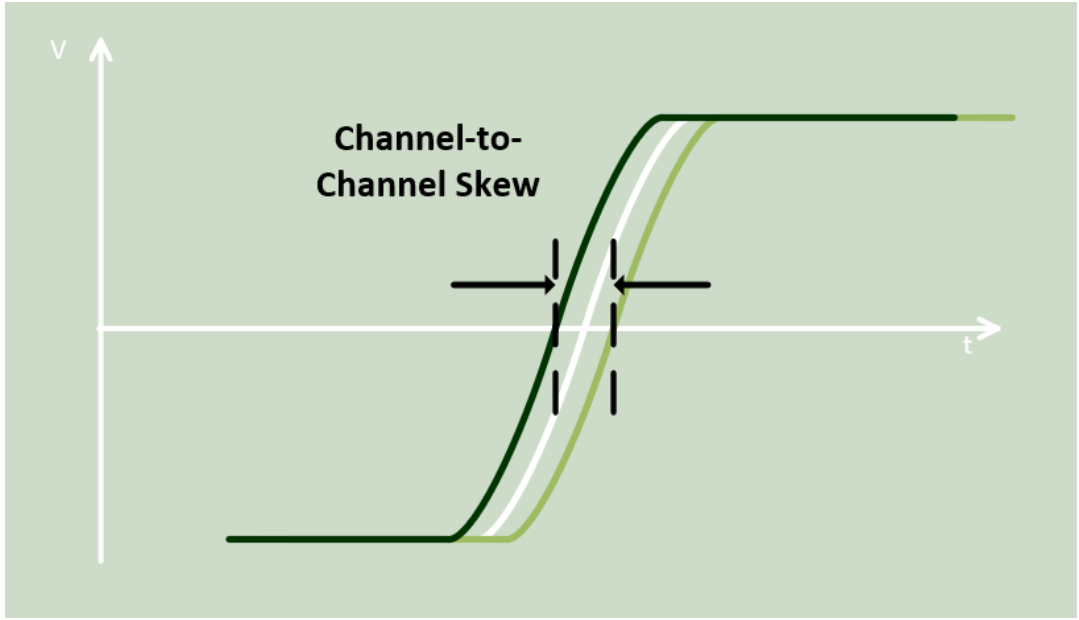
Drift

Two clocks, started at the same time, with the same frequency, will become unaligned over time.

Why?

- Inconsistencies in manufacturing
  - Two identical parts aren’t really identical
- Inconsistencies in part sourcing
  - Oscillators have tiny differences
- Physics at the nm scale are complex
  - Temperature!

Skew



A clock or trigger, sent by the same source, may become unaligned by the time it reaches two different destinations.

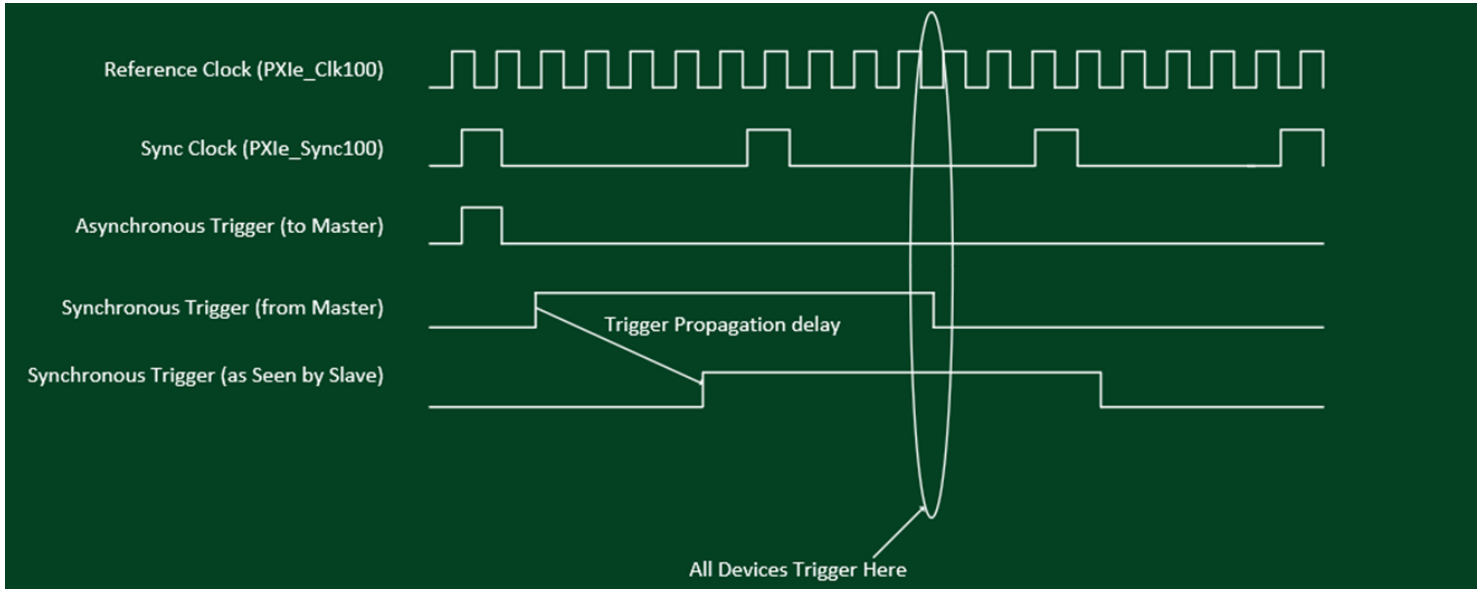
Why?

- Different signal paths!
  - Cable lengths, PCB trace routing, etc…
- Temperature
- Input Capacitance

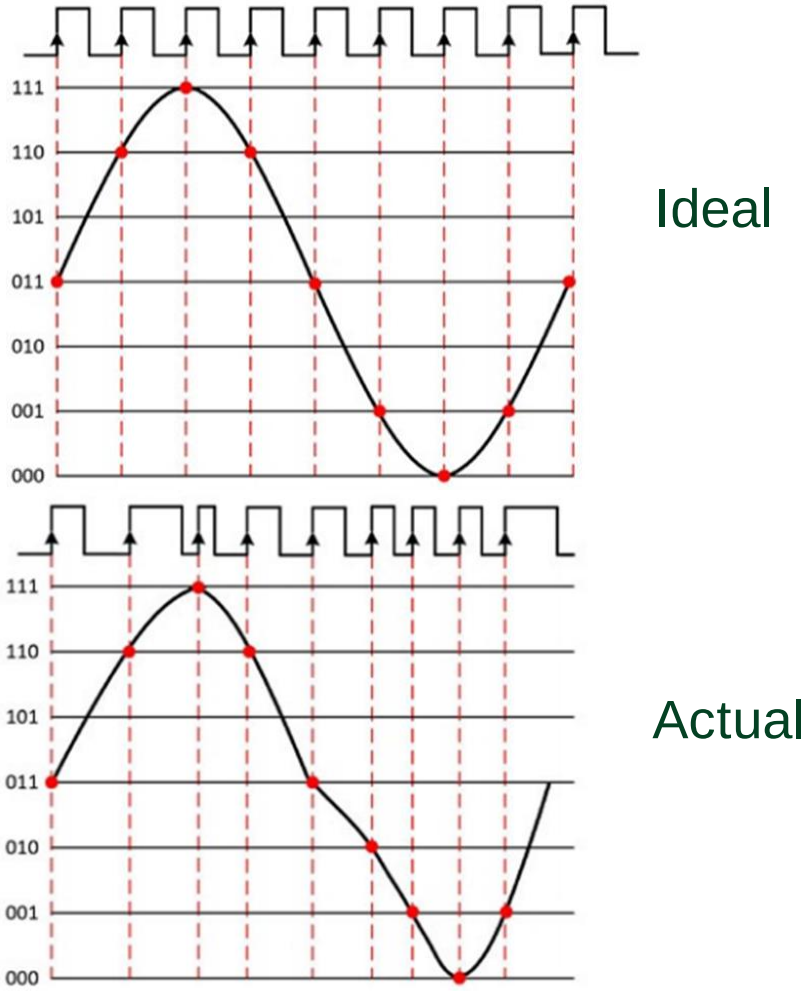
Trigger Skew Correction

An intermediate “master” device, optimized for trigger distribution, will correct for skew.

- Master device receives trigger
- Master device will calculate propagation delay.
- A new trigger will be distributed out to target devices synchronized to a reference clock.
  - Preferably distributed on an optimized pathway.
  - PXI Star lines



Jitter



Ideal

Actual

Deviation from the ideal timing of an event to the actual timing of an event. Often occurs as a pseudo-random event within a measurable window.

Why?

- Signal Crosstalk
- EMI
- Simultaneous Switching Outputs
- Other regularly-occurring signals

Learning Timing & Sync

How do we engage in the language of data acquisition?

Learning Timing & Sync

Literature: *Reading & comprehending the language rules*

- Hardware & API documentation

Exposure: *Immersing yourself in the experience of the language*

- Trial & Error. Figure it out and see what happens. Make mistakes.

Modern Tooling: *It's not the 1930s anymore. We have the technology!*

- Not all APIs are made equal.
- You don't need to do everything from scratch!

DAQmx

Allows for precise control over the timing behavior of acquisition and generation tasks for DAQmx-compliant hardware.

Targets: DAQmx-compliant hardware

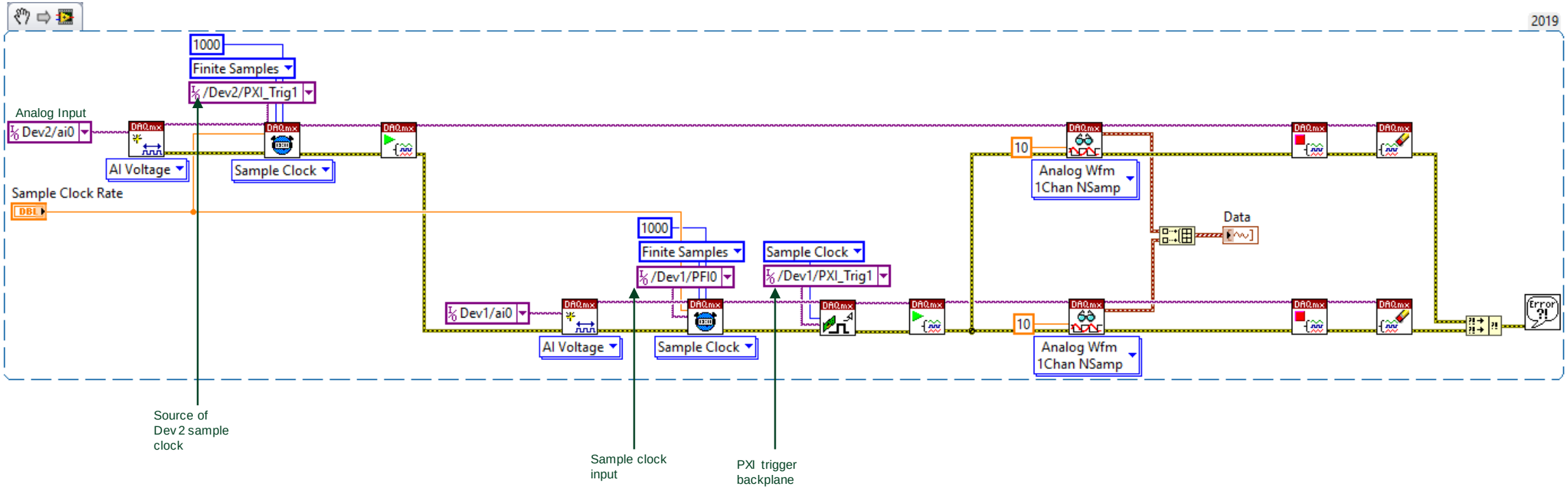
Use Cases: Single-card synchronization*

- Reference Clocks, Sample Clocks
- External Triggering, Backplane Triggering, Trigger Routing

* Can support multi-card routing within a single PXI chassis.

DAQmx Example

Sharing a sample clock between devices



NI Sync

Specializes in multi-chassis synchronization and specialty timing hardware.

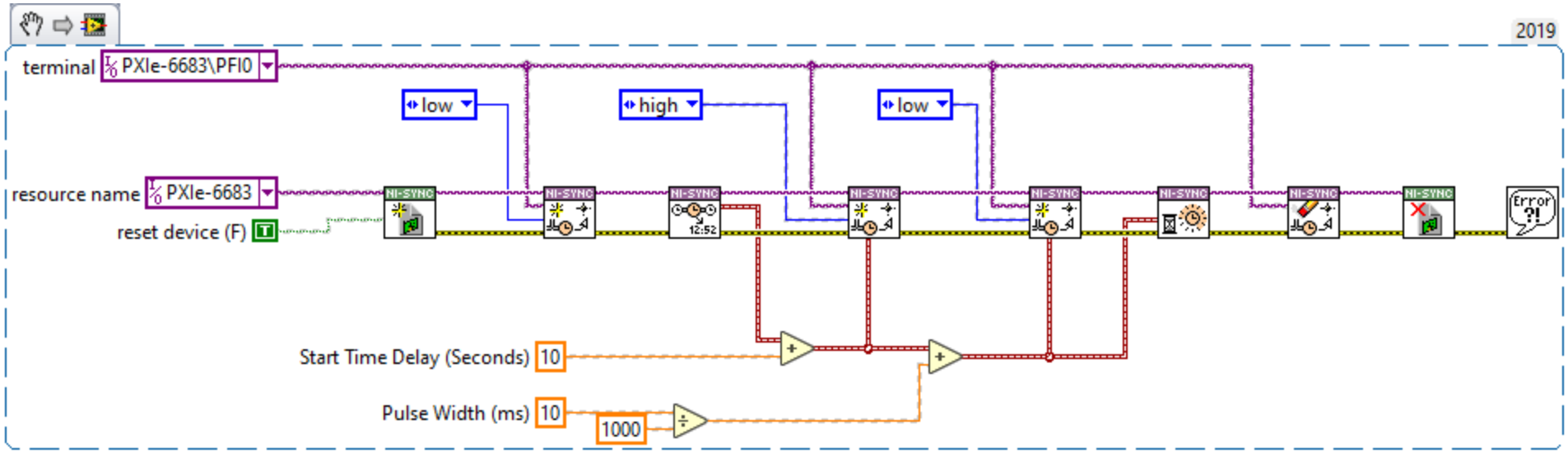
Targets: Multi-Chassis Systems, Specialty Timing Hardware*

Use Cases:

- Complex heterogeneous timing requirements
- Multi-chassis sync
- External timing sync (Signal-based or Protocol-Based)

*IRIG-B, GPS, 1588, etc… (PXI-6683, PXIe-6672, PXIe-6674T)

NI Sync Example



Scaling Things Up

The requirements are growing!

- Lots of Signals
- High Acquisition Rates
- Multiple PXI Chassis

Advanced Timing & Sync

High Speed Acquisition with NI-TClk

NI-TClk

A special API developed by NI to support synchronous high-speed data acquisition and generation.

Use Cases:

- Device clocks must be synchronous
- High Sample Rates
- Synchronous & Asynchronous Triggering
- Non-DAQmx
- Different Base Clocks

Key Terminology

Trigger Types:

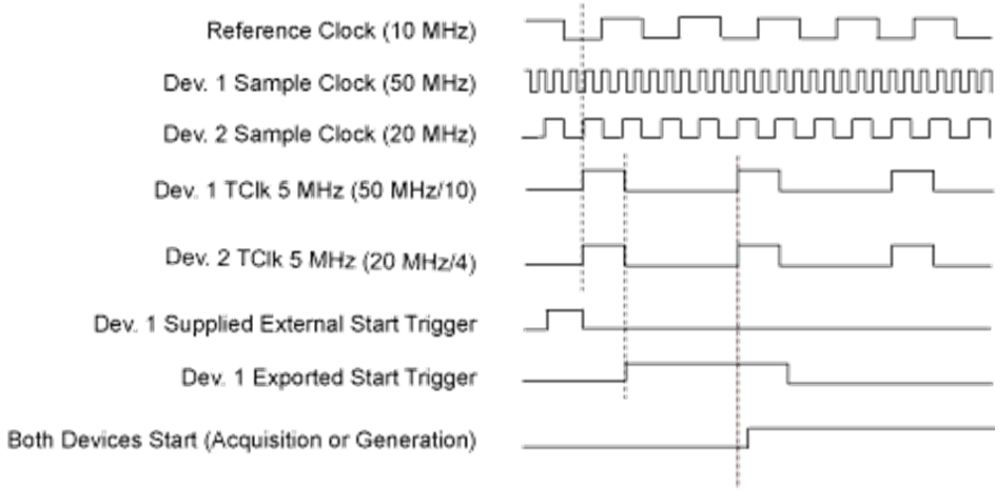
- Start, Reference, Script, Pause

Homogenous Triggering:

- Synchronous

Heterogenous Triggering:

- Non-Synchronous
- Unsupported by all devices
  - Otherwise unconfigured
- Different trigger sources



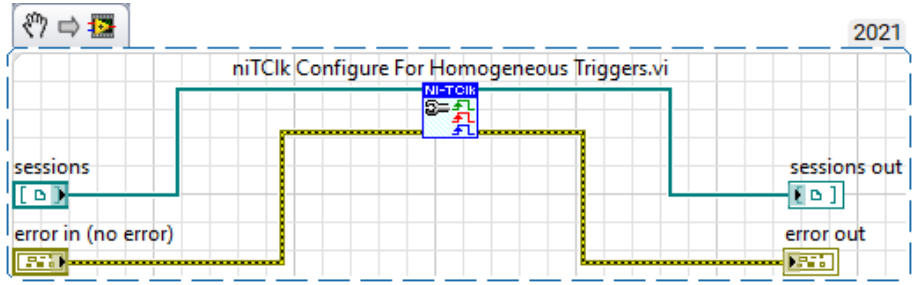
Core API Functions

niTClk Configure for Homogenous Triggers

Looks at all the input sessions and makes decisions

Configures the following:

- Reference Clocks
- Start Triggers
- Reference Triggers
- Script Triggers
- Pause Triggers



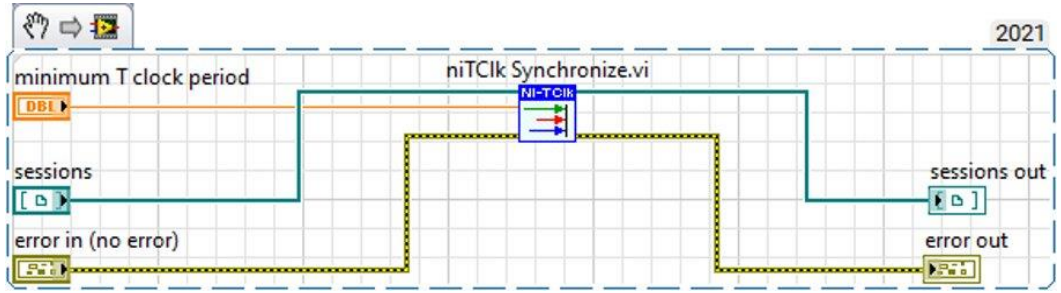
In a nutshell, prepares all the physical signaling for your application.

Core API Functions

niTClk Synchronize

Performs synchronization of all TClk signals

- Calculates local propagation delay
- Minimizes skew



The TClk period is adjustable

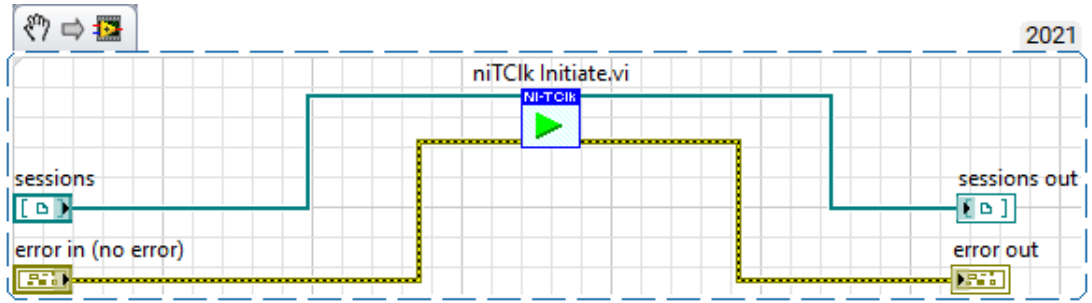
- Minimum of 200ns on a single chassis
- Useful when accounting for additional propagation delay in multi-chassis configurations

Core API Functions

niTClk Initiate

Final Configuration

- Awaits all low-level session init for sessions that share start triggers (for example)



Starts the TClk sessions

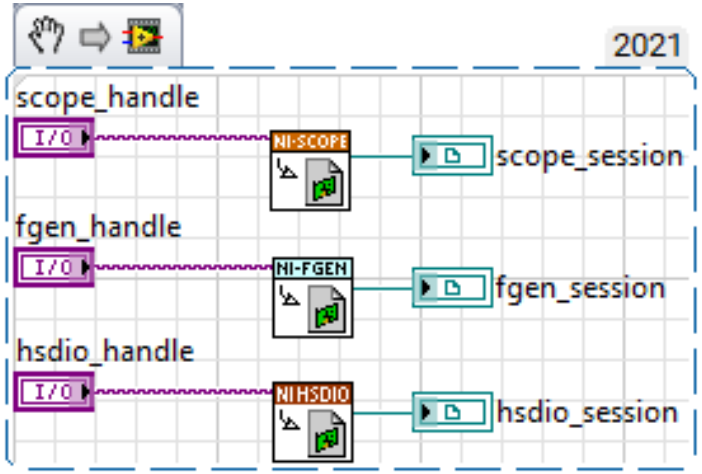
- Synchronous start time
- Synchronous sampling/generation

Native Session Functions

Get Session Reference

Session-Based Hardware

- Provides the session reference for NI-TClk to do its business.
- Available on a per-API basis
  - NI-FGEN
  - NI-SCOPE
  - NI-HSDIO
  - Etc...



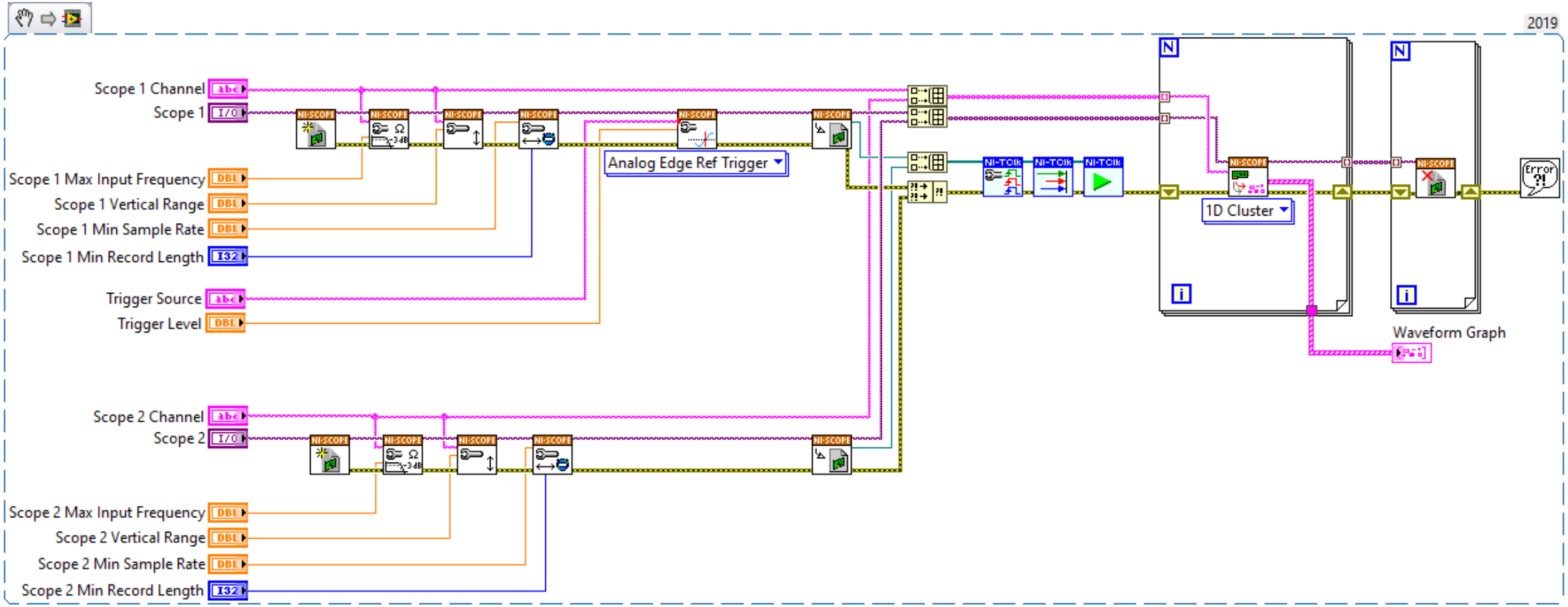
Note: Acquisition/Generation configuration and trigger setup will be done natively as well.

NI-TClk Operation

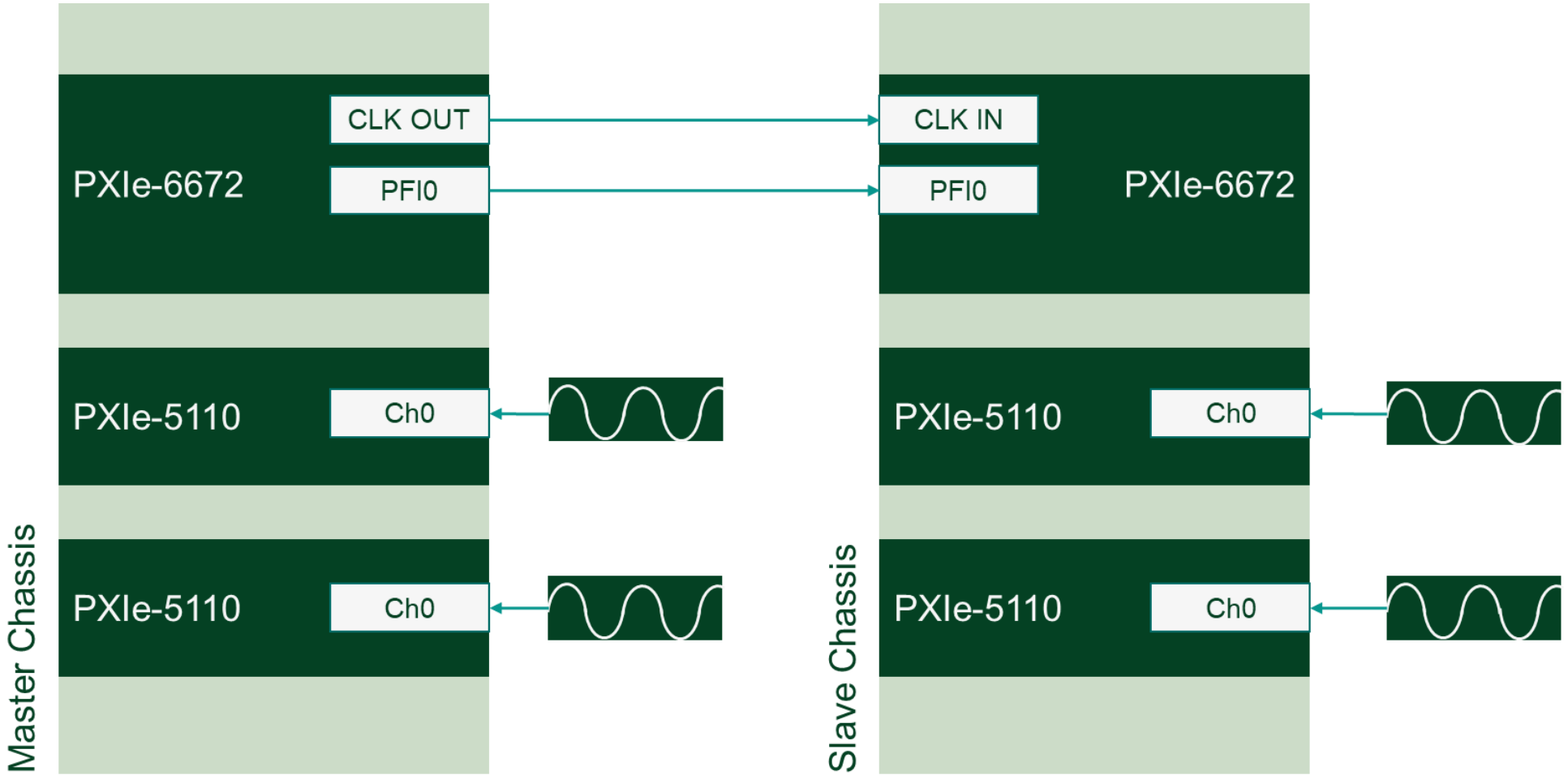
1	Devices start with initially unaligned sample clocks.
2	Each device generates its own TClk signal whose period is determined by the software based on the sample clock periods of all devices included in the TClk group.
3	Device sample clocks are phase-locked to the PXI 10 MHz reference clock which serves as the Sync Pulse Clock (or, optionally, to a high-precision externally supplied clock).
4	A Sync Pulse is generated, and each device waits for the first rising edge of the Sync Pulse Clock after its receipt.
5	The time between this rising edge and the first rising edge of the device's local TClk signal is measured, and the offsets are used to adjust device sample clocks and TClks to bring them into alignment.

Single Chassis Example

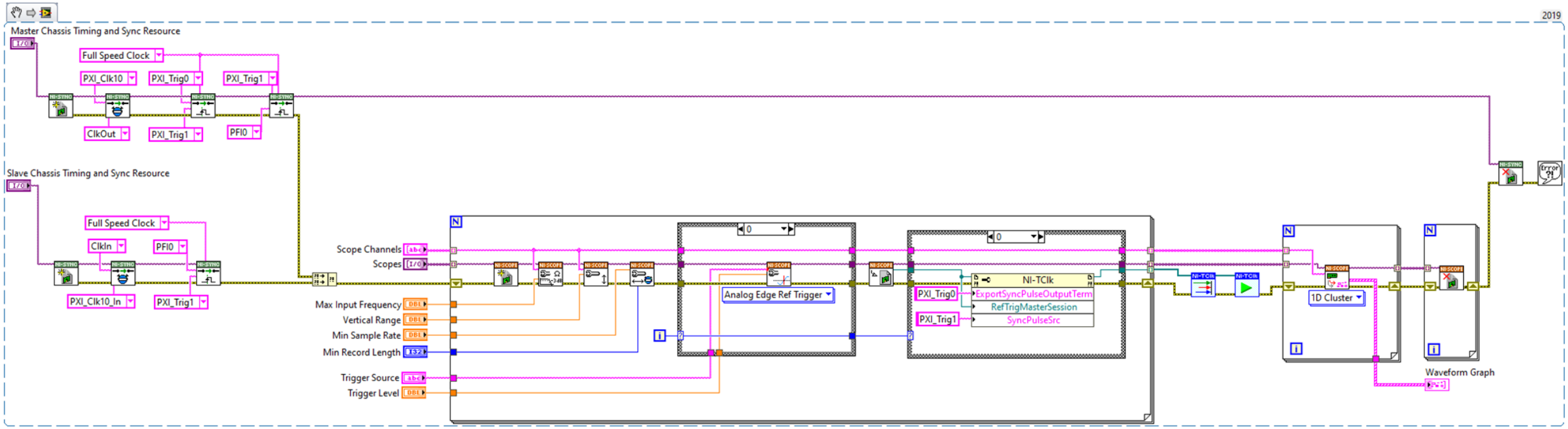
Synchronous



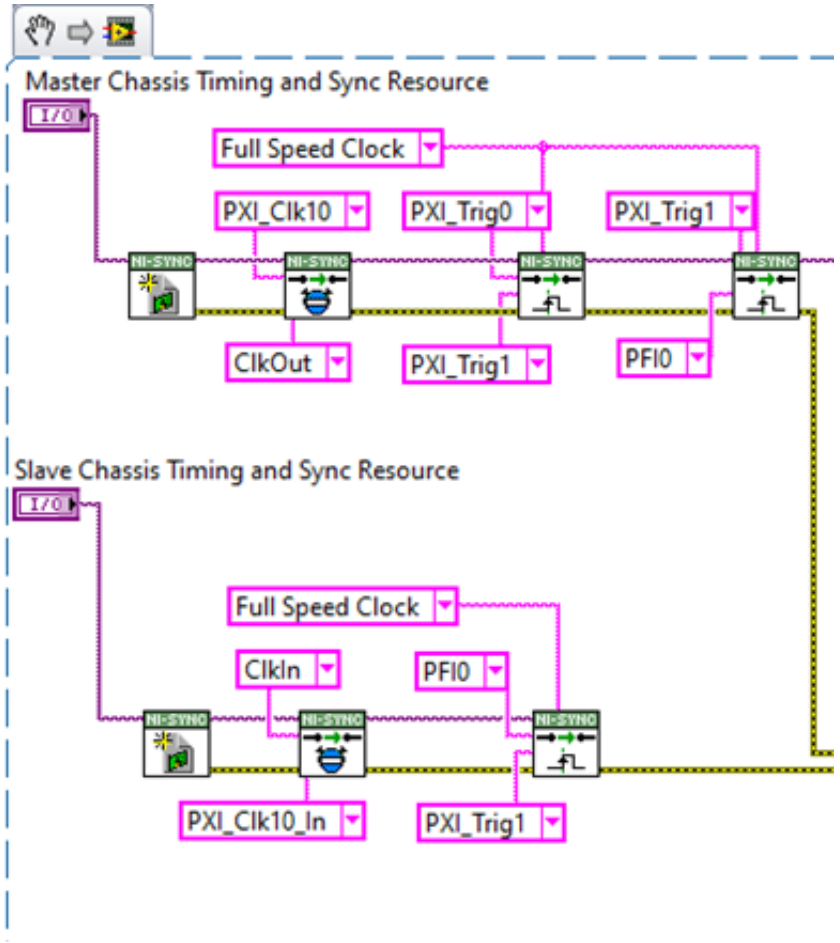
Multi-Chassis Synchronization



Multi-Chassis Example



Multi-Chassis Example



Case Study

50-Unit Synchronous Data Acquisition – Electronic Fuses

System Definition

The system shall provide a single RS485 bus to be distributed across 50 DUTs
- Each DUT shall be powered individually and shall respond to its address within the RS485 command

Each DUTs output shall be routed to an independent scope channel

Upon receiving a “Fire Command” the DUT shall respond by generating a 2us pulse

The Problem

The system must evaluate the timing between the RS485 command and the 2us pulse output to ensure the DUT responded to the “Fire Command” in the allotted amount of time.

- Presence of pulse
  - Used FPGA to verify no pulse on non-target channels
- Characteristics of the pulse
  - Rise time, slope, falling time, area under curve
  - Care about the energy released in pulse
  - 60MS/sec
- Time from Command to Pulse generation
  - Happened within deadline

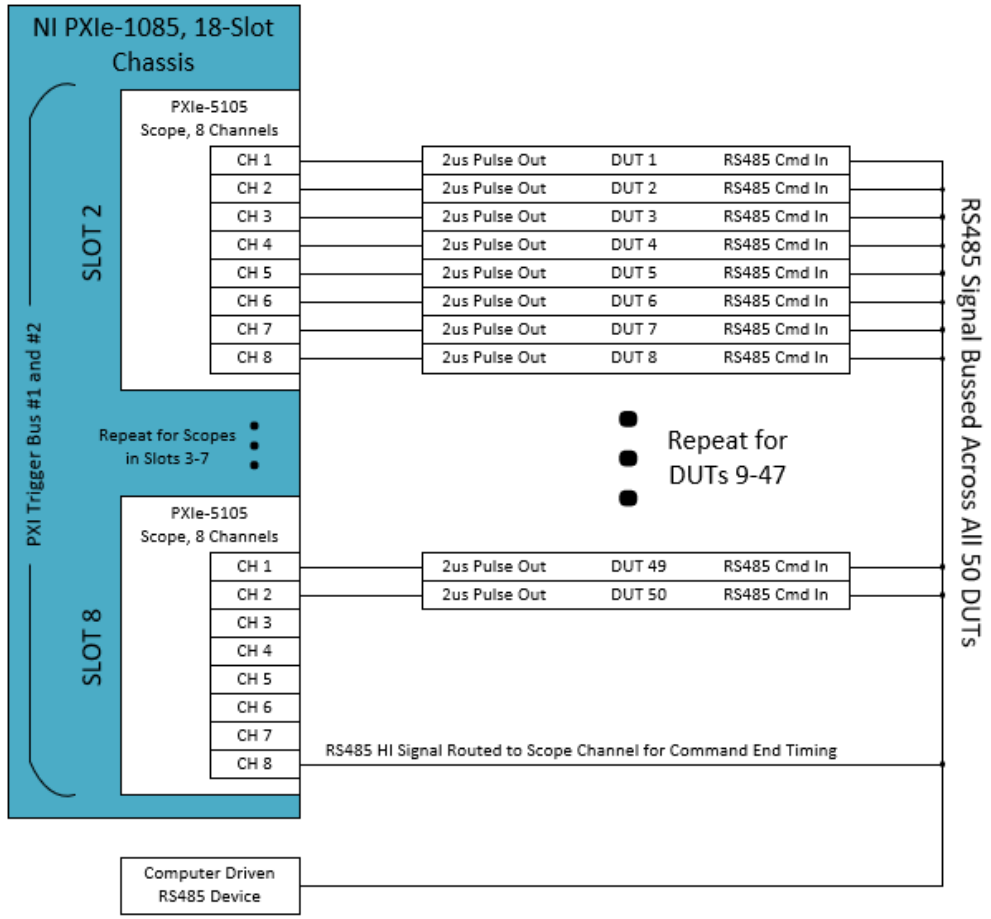
Hardware Implementation

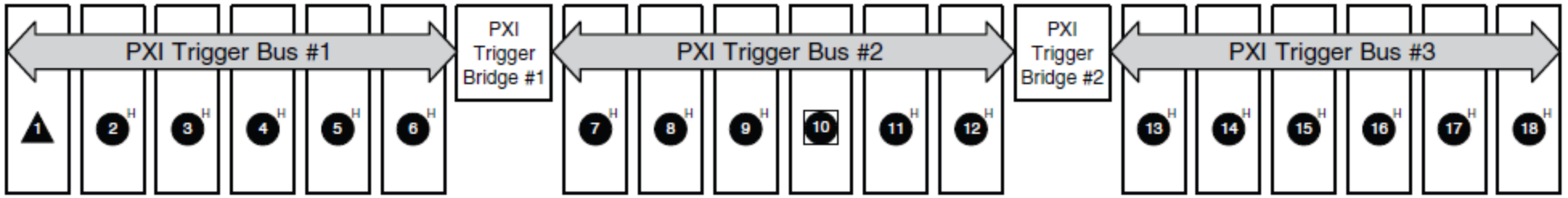
(Quantity 7) PXIe-5105 8-channel scopes were used to provide an independent scope channel per 50 DUT pulse outputs

RS485 commands are sent by a computer driven device and distributed across all 50 DUT command inputs

The RS485 HI signal was routed to CH8 on the scope in slot 8 to provide capture of the RS485 command

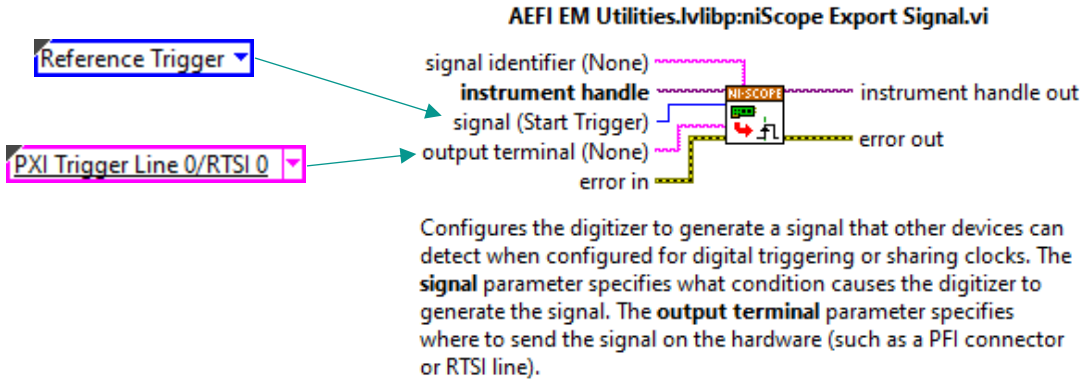
The PXIe-1085 chassis required a bridge between trigger bus #1 and #2 to share the RTSI_0 trigger line



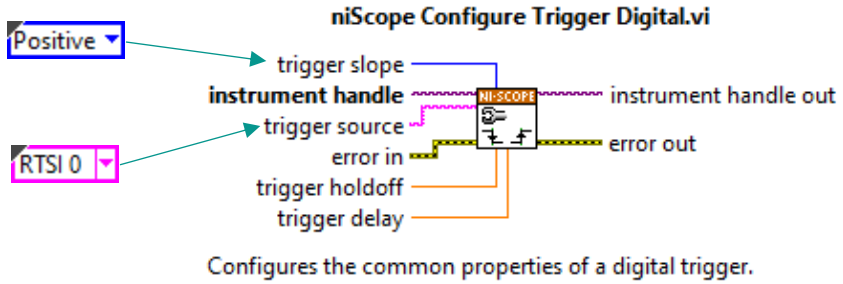


Software w/o NI TClk

Configure scope channel for one of the DUTs to trigger on input signal and to export signal on RTSI_0



Configure scope channel for the RS485 HI signal to trigger on a positive slope of the RTSI_0 trigger source



Arm both the DUT pulse output scope channel and the RS485 HI signal scope channel

Send the “Fire Command” addressing the designated DUT over the RS485 bus

Fetch the data for both channels

Analyze the RS485 command data for the fire command to fire pulse timing

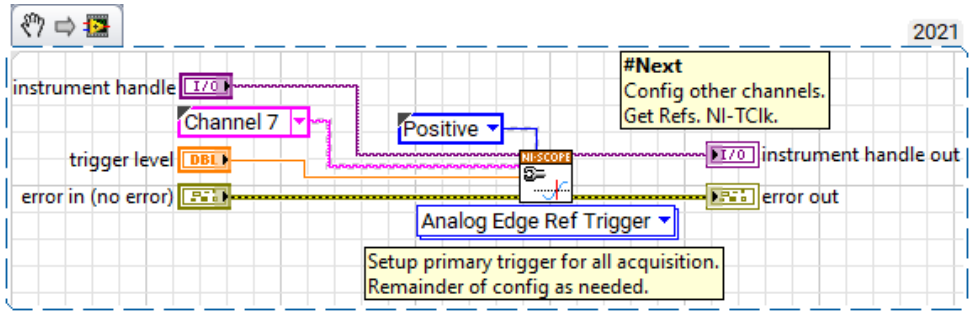
- The center of the RS485 capture is the instance the fire pulse was received
- The timing from this to the first data bit sent is the timing required

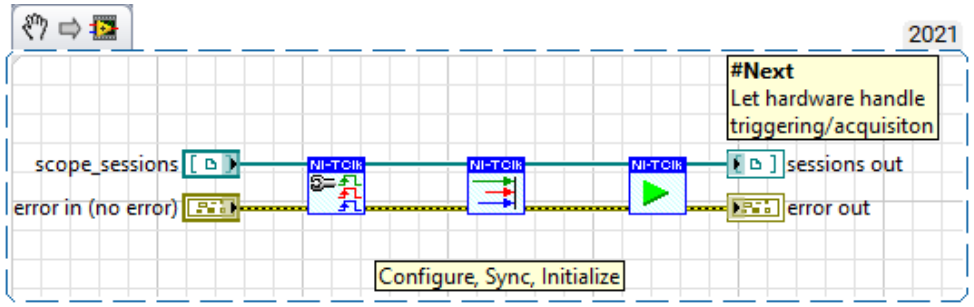
Software w/ NI TClk

Setup PXI hardware for general use. Do not need to manually bridge triggers.

Configure all scope channels for acquisition. Configure one scope channel to await trigger conditions. Do not route triggers to RTSI lines.

Call TClk Homogenous Configure, Synchronize, and Initiate. The system is now armed. All channels will use the same trigger conditions.



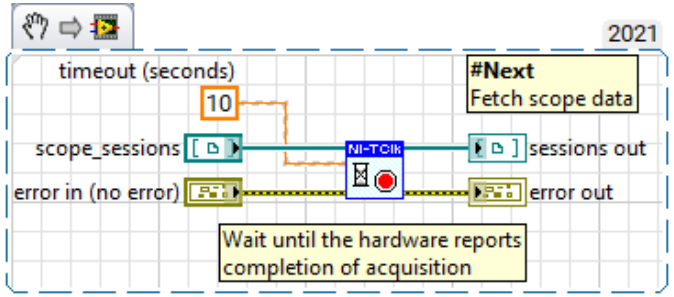


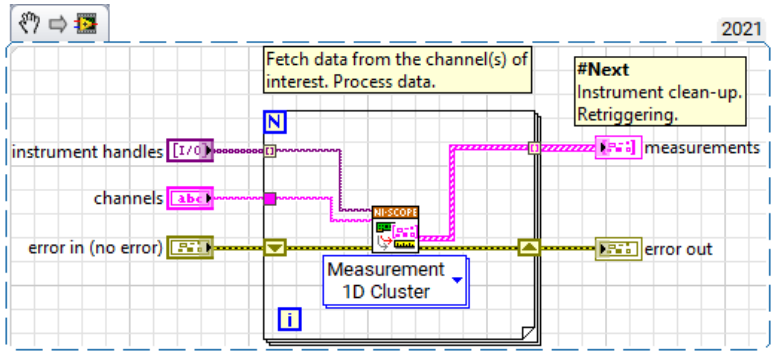
Software w/ NI TClk

Use the TClk "Wait Until Done" VI to pend until the acquisition is complete.

Fetch data from the scope channels.

Clean-up instrument references. Reconfigure as needed. Reuse existing code for retriggering.





Results

Reduced system configuration

- No need to manually bridge backplane (minimize downtime)

Improved scope channel synchronization

- Data points are less affected by drift/skew

Improved scalability

- Simple to add further cards and channels, reuse configuration

Improved system capabilities

- This is an R&D system
- More complex testing is available

Closing Thoughts

Timing & Sync

Use Timing & Synchronization Concepts to improve your data

- DAQmx provides simple tooling for routing and sync
- Use NI Sync for applications with more strict requirements

Leverage NI-TClk to reduce complexity & improve results

- Minimize manual configuration
- Reduce physical artifacts in synchronization
- Improve scalability
- Synchronize multiple PXI chassis systems

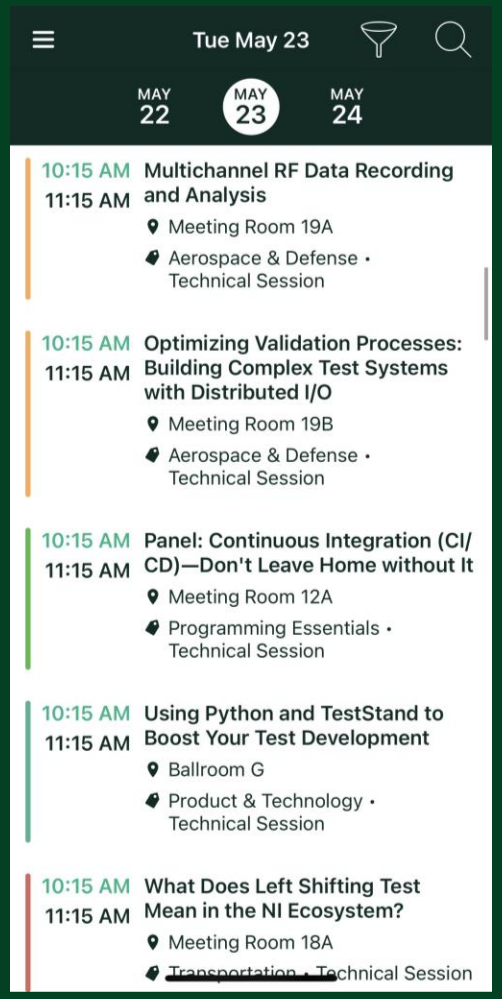
Useful Links

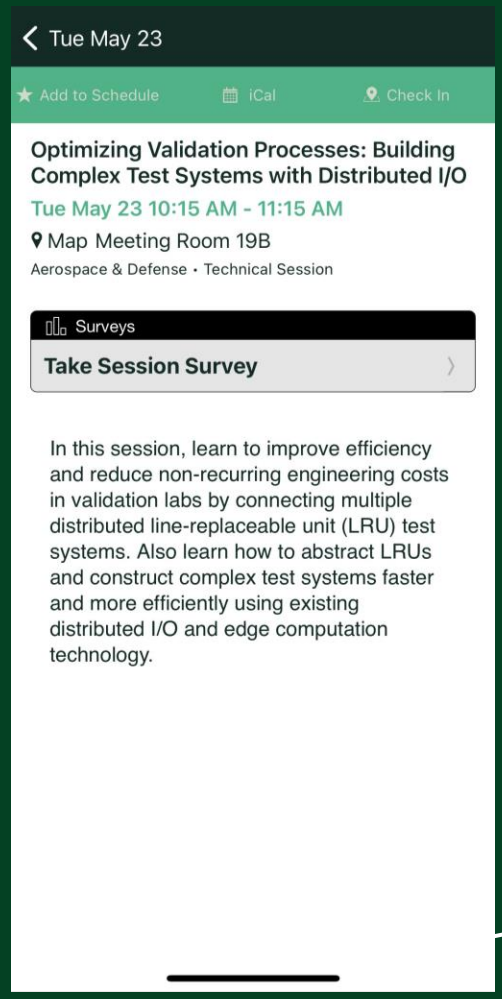
- DAQmx Documentation
- NI Sync Documentation
- NI TClk Documentation
- NI Official Timing & Sync Training Course

Questions?

Give us your feedback!
Quick 2 Question Survey

In the mobile app, click into the session you would like to provide feedback for





Click “Take the Session Survey”