SAND2019-2526C

Title: Simple Flexible Sequencing for Complex Coordination

Title: 65-character maximum

Challenge Statement: 1,000-character maximum

The needs of Research and Development facilities are complex and ever changing. Building a control system that can accommodate the needs of many while still being maintained and operated by a few personnel is a challenging problem. Z-Beamlet, the prototype pulsed power laser for the National Ignition Facility is living its second life as a diagnostic for the Z Machine while also being used for independent pulsed power laser experiments. The original control system that was built to operate the prototype laser was hampered by race conditions and unreliable hardware. The operators would spend their time during the countdown hoping things went well and deciding on whether to abort or not before the system failed. Over the years, additional pulsed power systems were added to the facility which included a second laser with a higher power rating (Z Petawatt), a smaller pulsed power laser (Chaco), and a pulsed magnetic field (Bfield). These 4 systems, their many support subsystems, and the Z Machine are called on regularly to carry out a variety of high energy physics experiments that often involve coordinated events utilizing one or more of the various pulsed power systems. Because these experiments are all single-shot events, they rely heavily on high reliability and deterministic outcomes. These things become even more complicated when you're dealing with heterogenous, asynchronous, distributed systems spanning multiple decades, multiple vendors, and multiple buildings.

Control systems engineering resources are very limited (one person) so solutions couldn't require complex modifications. Everything had to be simple, straightforward, flexible, and reliable to reduce troubleshooting time when hardware failed. Interfaces for the different subsystems needed to be easily and independently testable since aging hardware often fails in new and interesting ways. Subsystems are spread over a large area so being able to quickly see that a problem exists before attempting a shot event is a high priority to keeping operations moving forward. Additionally, scientists rarely know what they want before it's built, so the less time spent bringing up a new system and integrating it, the quicker the final subsystem can be implemented/re-implemented.

Please consider the following in order to craft an award-winning challenge statement.

• What challenge (business, industry, or societal) did you need to overcome?

• Why was it so important to address your challenge?

• What tools/methods did you use for your application before NI products?

Application Overview: 2,000-character maximum

"Shot Control 2.0" was intended to address the issues inherent to the old system while allowing for future expandability and increased complexity without sacrificing simplicity and reliability. LabVIEW and NI Hardware (including PXI, cRIO, and single-slot cDAQ) were selected to tie everything together into one extremely flexible, coordinated, easy to operate, and easy to maintain system. LabVIEW was selected because it can quickly provide a usable GUI for troubleshooting, it can easily interact with a wide range of equipment (RS485-based Optimux I/O Boards, Oscilloscopes, Delay Generators, High Voltage Power Supplies, Aerotech Motion Controllers, Calorimeters, PLCs, Valves, Sensors, PA systems, Pumps, Triggers, etc.), it can be easily deployed to desktops, servers, and embedded controllers, and it

can even be used to program FPGAs without needed to learn a new language. NI Hardware (including PXI, cRIO, and cDAQ) was chosen since it covers a wide range of interface needs and they integrate easily with LabVIEW. Digi One SP and Prologix GPIB-ENET modules were selected to integrate RS232/RS485/GPIB devices to the network due to their low cost, simplicity, and reliability. For a one-man team, these things really matter.

To solve the complex operation issues in coordinating 4 to 5 pulsed power systems, in various combinations, flexibility is key. The primary parts of the Shot Control 2.0 project include:

- Sequencing Engine
- Sequence Generator
- Shot Configuration
- Independent Subsystems that interact with the Sequencing Engine
- Watchdog
- Timing/Triggering System (Heartbeat)
- Program Management Launcher

The primary component of the Shot Control program is a basic sequencing engine. The events handled include “Send Command to Subsystem”, “Wait for Subsystem to be Ready”, “Tell Watchdog to Watch/Stop Watching Subsystem”, and “Abort Sequence”. The sequence engine processes one event at a time and either continues to the next event, rechecks the event until a timeout elapses, or aborts the shot due to a failed condition check. Since all events are logged programmatically, investigating failed shot attempts is fast and easy with conclusive results pointing towards simple fixes such as fixing/replacing failed hardware, modifying subsystem code, or adjusting timing. More complicated events can easily be added to the short list of events within the sequencing engine but are often not needed.

This automation sequence utilizes Shot Sequence files that are generated based on Recipes that create subsets of the Master Shot Sequence file. The events within the Master Shot Sequence file are grouped together to help make the Recipes a simple matter of selecting the type of events that are desired within a given Shot Sequence. This allows the system designer to easily integrate new subsystems with their own sets of events into the overall flow of the sequence without too much hassle. Changes to the Master Sequence are reflected in the individual Shot Sequences by using the known Recipes to automatically regenerate all requested sequence combinations. Creating new Recipes is as easy as selecting which groups of events should occur for an individual Shot Sequence.

Shot Configuration parameters such as power supply selections, charge voltages, and sensor characterizations are in a separate file that is selected by the operator before the shot. This information is relayed to the individual subsystems at the beginning of the shot sequence to convey details of how to properly implement a command or how to interpret data during the event (for display, logging, or decision purposes).

Individual subsystems are initially sent the Shot Configuration containing subsystem parameters at the beginning of the sequence. For example, when a system is told to CHARGE, it would have been previously sent information that included an argument for a “Charge Setpoint” of 18kV. After the sequence engine has confirmed a subsystem to be READY, it can then be told to be CRITICAL before the sequence engine moves on to something else. Being CRITICAL gives the independently running

subsystem the power to abort a Shot Sequence (with a reason) if something unexpected occurs after it was previously deemed to be READY.

In case of a subsystem falling offline unexpectedly, the Watchdog will notify the main sequence engine that it has lost communication with one of the systems it was directed to watch during the sequence. This covers the gap between delegating to a subsystem that it has the power to independently ABORT a shot but then not being sure if said subsystem is still functioning. It also serves as an easy check of which systems are functioning properly before a shot sequence (or anytime) so that repairs can be started long before a shot is attempted.

The final critical piece of any single event system is that of the Timing and Triggering. This hardware system operates throughout the shot by providing trigger pulses at 0.2Hz (5 second interval) which are received by the various subsystems that rely on precise timing. This also allows the automation a 5 second window to arm all relevant subsystems for the final event trigger. The presence of this 0.2Hz signal is monitored for stability throughout the shot sequence as an added check to ensure a reliable trigger will occur when the time comes.

Maintaining a complex, distributed system like this with only one control systems engineer is only achievable when making updates and adding additional subsystems take minimal effort. Towards this effort, a lot of helpful pieces have been added to the system over the years that allow rapid deployment and easy testing.

The Rapid Deployment and Launcher System operate in two stages. First, after building an executable from within the LabVIEW project, a Post-Build Action will automatically copy the executable to a timestamped folder on a network shared folder. Then, when a computer (laptop, desktop, server, or PXI) that is part of the control system is told to run Shot Control programs, a launcher will check a pre-defined list and run the latest version of each program needed. In this manner, a complete re-deploy of 30+ individual executables across 7 systems can be accomplished with a few mouse clicks. cRIO-based systems are redeployed directly from the project to the intended cRIO controller.

cRIOs were selected for controls due to their simplified maintenance as well as their flexibility. Single-slot cDAQs were selected when a remote sensor needed to be measured and recorded without any of the complexity of a cRIO. PXI chassis were used in cases of high I/O count, but they haven't aged well due to hardware failure and software corruption, possibly related to EMP effects. For our purposes, PXI controllers will most likely be replaced with modern cRIO units when the opportunity presents itself.

NI's product suite forms a convenient ecosystem, allowing a single Control Systems Engineer to tackle large, complex tasks and have a reasonable hope of maintaining such a system for the extended lifespan of large, high energy physics machines.

Please consider the following in order to craft an award-winning application overview.

• Describe your application and how it works.

• Which NI products are you using (software and hardware)?

• What obstacles did you have to overcome and how did you overcome them?

• Explain why you chose NI technology to develop your application over other options?

• How is your application better because of NI products or the NI ecosystem?

Impact Statement: 2,000-character maximum

The new control system architecture has enabled major changes to be quickly implemented while minimizing overall system downtime. These changes have recently included multi-user control from various locations, upgrading subsystems from legacy to modern hardware, and adding new support subsystems to existing shot sequences. More complex shot sequences are being requested that involve multiple pulsed power systems firing synchronously into more complicated test chambers. Requests for additional system capabilities are now implemented in a reasonable time frame without requiring extensive testing.

The largest impact of the new control system architecture has been in enabling more reliable shots with the Z Machine. The Z Machine has a unique capability to carry out certain types of high energy density experiments. Z shots are valuable both for the data generated but also in operating costs ($200k-300k per shot). Four to five times more requests for experiments are received annually than the shot schedule allows which makes the value of the shots even higher. More reliable systems lead directly to better quality data and more of it.

The biggest lesson learned is that keeping things simple allows for greater flexibility when things need to change. Overall system complexity can be handled by simple objects which can be simply implemented, tested, repaired, and integrated together to make complicated things happen. LabVIEW's ease of use and troubleshooting complemented NI's ease of hardware integration which ensured that all of the simple things could simply work together.

Please consider the following in order to craft an award-winning impact statement.

• Explain the benefits, preferably quantifiable benefits, of your application. For instance, how much time, money, or complexity was saved?

• What is the impact of your application? This could be an impact on your business, your customers' business, society, an industry, etc.

• Why is this application significant? Was it the first of its kind?

• What did you learn or discover?

• What feedback have you received? Have you won any awards or received any accolades? Has your application been published?