



SoLID Slow Controls

Brad Sawatzky

June 8, 2020

Summary

- Slow controls for project on this scale is significant
 - Hall B → 2+ FTE (professionals) for ~2 years (6 people made significant contributions)
- Think about and document slow control needs
 - Feed your requirements/design specs to <brads@jlab.org>
 - I'm happy to support research and answer questions
 - Everything will be more \$\$\$ and more complicated than you may expect...
- Standardize, standardize, standardize
 - Avoid investing time in 'quick' solutions for local implementation. Stick with the standards – steeper learning curve, but it'll save time in the long run (build trained people as well as software).
 - Hacks and workarounds tend to become 'permanent' and unintended dependencies get baked in – good to avoid these.
 - Proper hardware selection will minimize custom IOC/PLC development.
- EPICS should be our common API/Protocol
- Frontend GUIs/software take time and \$\$ to develop
 - Control Systems Studio (CSS) framework is recommended
 - Software maintainability is *at least* as important as features/functions

Detectors – General Requirements

- HV / LV controls, Temperature, Pressure GUIs with **EPICS compatible logging (and alarms)**
 - Appropriate crate selection makes this straight forward. Recommended systems have control, monitoring and alarm loops already implemented, no IOC/PLC development needed.
- LED Gain monitoring (“on/off”) remote controls are straight forward
- “Flow-through” / open-loop gas systems (GEM, LGC)
 - **Solved problem with pre-existing GUIs. Go with a standard MFC, etc.**
- Heavy Gas Cerenkov gas system
 - Infrastructure can be complex, but slow controls are minimal since fills are done manually (and rarely) by an expert, then system is sealed during production.
 - Just needs online monitoring of pressures, temps—fairly straight forward
- Only “fast” interlock that crosses (sub-)system boundary is to trip HV if GEM flow stops. Straight forward with recommended HV systems.

Frontend GUIs

- EDM (MEDM) / JTABS

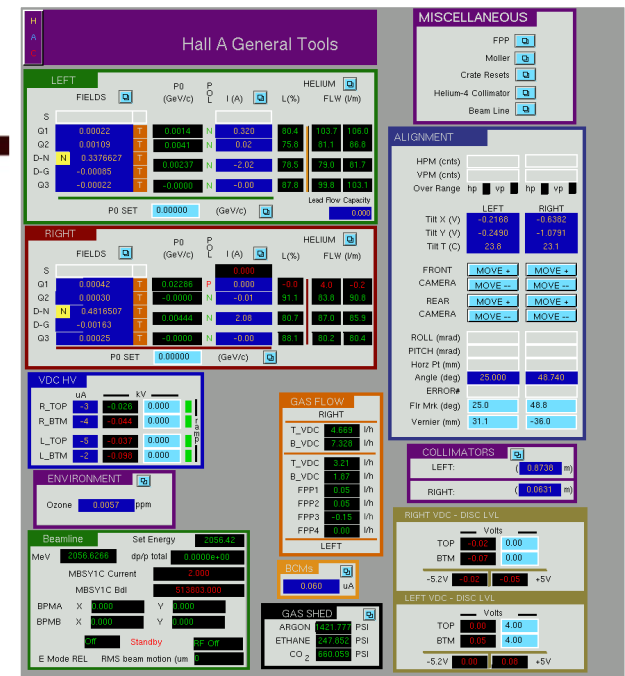
- Forward-port of JLab's 6 GeV EPICS screens
- Still developed, but dated

- Control Systems Studio

- <http://controlsystemstudio.org/>
- Eclipse-based toolkit designed for systems like ours
 - SNS, BNL, FRIB, DESY using this system
 - JLab: Hall D (in use), Hall B (in use), Hall C (in use)
- Now migrating to [Phoebus](#) (replacing Eclipse UI framework; same idea)

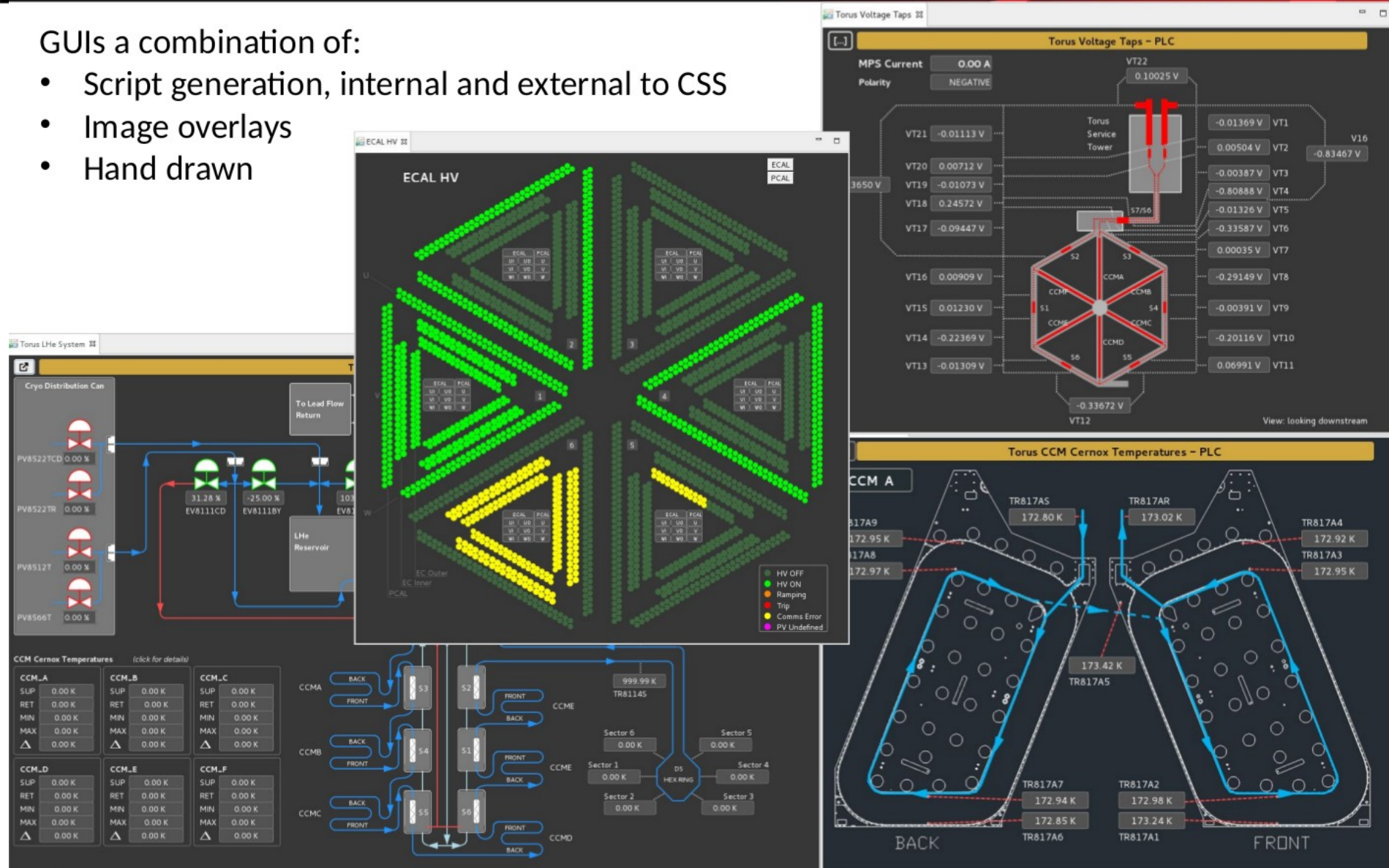
- Will enforce standards across systems

- Avoid LabView
- Avoid custom/proprietary code as much as possible
 - if not possible, provide EPICS interface for integration



GUIs a combination of:

- Script generation, internal and external to CSS
- Image overlays
- Hand drawn



Slow Controls System Overview

Detector	HV / LV Power	LED flasher/pulser	DAQ Crate Monitoring/Control	Gas System Type	Temp Monitoring	Flow	Pressure	Fast Interlock	Comments
GEMs	x		x	Flow through		x		x	75/25 Ar:CO ₂ mix; HV interlock w/ flow
LA/FA SPD	x	x	x						
ECal	x	x	x						
Light Gas Cerenkov	x	x	x	Flow through		x			1 atm(abs); CO ₂ , N ₂
Heavy Gas Cerenkov	x	x	x	Fill & Seal	x	x	x		1.5–1.7 atm(abs) C ₄ F ₁₀ or similar

- HGC gas system design is not well defined
- MRPC is not listed here (out of scope)
 - MRPC gas system on this scale is likely a big deal

Summary (Slow Controls)

- Even with component standards enforced, and fairly modest requirements, slow controls for project on this scale is still significant
 - Hall B → 2+ FTE (professionals) for ~2 years (6 people made significant contributions)
- Standardization and cross-system oversight is critical prior to purchase to avoid issues
 - Direct/Authoritative oversight by CAM?
 - Ensure EPICS and other low level interface support is present and to spec
 - Avoid home-built and proprietary software where possible
 - Identify and communicate system needs that may cross sub-system boundaries
 - EPICS will be our common API/Protocol
- Maintainable Frontend GUIs/software require sufficient time *and* professional software developers
 - Control Systems Studio (CSS) / Pheobus framework is recommendation

Backup Slides

EPICS

- Experimental Physics and Industrial Control System
 - <http://www.aps.anl.gov/epics/>
 - Open source, actively developed, lots of users
 - Based on C; APIs available for Java, Python, LabView, etc...
 - Covers both input/output controllers (IOCs) that do the real work
 - *ie.* poll for and respond to data in real time
 - publish data for other systems to consume
 - IOCs can be single board computers running vxWorks, embedded devices that support the EPICS protocols, or 'softIOCs' which are applications that can run under conventional OSes (linux, etc)
- Main slow controls 'backend' used at JLab
 - A lot of expertise in Accel Div. that we can leverage
 - However, we need to schedule (and budget for) the developer time well in advance!
 - Archiving of slow controls data can be integrated with existing (Accel) MYA Archiver