Fermilab LLRF Controls Integration

Pierrick Hanlet
PIP-II LLRF Preliminary Design Review

9 September 2021

A Partnership of:
US/DOE
India/DAE
Italy/INFN
UK/UKRI-STFC
France/CEA, CNRS/IN2P3
Poland/WUST
Fermilab Control System Content

• Introduction

• Motivation

• Existing system
  - History
  - Integrating SLAC/LBNL software with Fermilab software infrastructure

• Proposed software restructuring

• Client software
  - edm -> Phoebus
  - Archiver
  - Alarms
  - save/restore

• Future consideration

• Summary
Introduction

- At Fermilab since 1989 as visiting scientist; joined staff in 2018
- 15+ years of experience using EPICS
- Working in Accelerator Division’s Front End controls team
  - Leading development and implementation of EPICS infrastructure at Fermilab
  - Goal is to simplify deployment of IOCs by non-experts
  - Goal is to modernize the client-side tools for operators in a seamless transition
  - Use modern computing methods such as Continuous Integration/Continuous Deployment (CI/CD) for code management

- Brian invited me to assist the LLRF team to support SLAC/LBNL software at Fermilab
- Recognize the value of the strong inter-lab LLRF collaboration
  - My task is to integrate the existing software into our framework **without breaking it**
Structure follows conventional EPICS implementations

The Fermilab “standard deployment” of EPICS code assumes a 3-tier build:

- **EPICS base** – main core of EPICS, comprising the build system and tools, common and OS-specific interface libraries, Channel Access and PV Access client and server libraries, static and run-time database access routines, the database processing code, and standard record, device, and driver support. Production code resides in `/usr/local/epics/base`

- **EPICS Support** – contains modules which are analogous to drivers one might add to the kernel for a computer to run specific functions and/or hardware drivers. We presently support ~50 support modules and expect this to grow. Production code resides in `/usr/local/epics/Support`

- **EPICS IOCs (Input/Output Controller)** – specific front end servers for controls and monitoring; these are built by pulling in Support modules and adding application specific code. A template IOC is provided to developers and already has minimal Fermilab required functionality. Production code resides in `/usr/local/epics/iocTops`

- EPICS base, Support, & iocTops are built for different architectures/platforms
- EPICS base, Support, & iocTops are hosted by NFS server
- Code base is built and tested in Continuous Integration/Continuous Deployment processes
- Goal is to have a robust EPICS code base to simplify novice EPICS-developer deployment and to simplify experts who help in debugging
- Builds exist for linux-x86_64, arm/Cyclone V, arm/Arria-10, arm/RasPi2, arm/RasPi4
Introduction – Software Path to Deployment

Fermilab’s EPICS Software Flow

Continuous Integration/Continuous Deployment

Development Nodes
- epic/epicsDEV/base
- epic/epicsDEV/Support
- epic/epicsDEV/Config
- epic/epicsDEV/locTopgs

Buildroot CL/CD
- addlinux
  - /usr/local/epics/base
  - /usr/local/epics/Support
  - /usr/local/epics/locTopgs

Built for:
- limit-rtb64 mvm8100
- Archit Soc Terasic SoC
  - RaPi 2, RaPi 4

Embedded IOCs built for:
- mvm8100
  - Archit Soc Terasic SoC
  - RaPi 2, RaPi 4

Unit Testing
- addlinux

EPICS unit tests
- Custom unit tests

Acceptance Testing
- Run Template IOC
- Linux node
  - on board

PV Naming Tests
- IOC

NFS Host
- epics
  - /usr/local/epics/base
  - /usr/local/epics/Support
  - /usr/local/epics/locTopgs

FE Servers
- NFS mount
  - Raspberry Pi
    - /usr/local/epics/base
    - /usr/local/epics/Support
    - /usr/local/epics/locTopgs

  - IOC Linux Host
    - ioc01
      - /usr/local/epics/base
      - /usr/local/epics/Support
      - /usr/local/epics/locTopgs

  - IOC Linux Host
    - iocNN
      - /usr/local/epics/base
      - /usr/local/epics/Support
      - /usr/local/epics/locTopgs

  - Embedded Controller
    - ioc
      - /usr/local/epics/base
      - /usr/local/epics/Support
      - /usr/local/epics/locTopgs

Clients
- NFS mount
  - EPICS Console
    - Phoenix
      - /usr/local/epics/Config
      - /usr/local/epics/Extensions

  - EPICS Console
    - Phobos
      - /usr/local/epics/Config
      - /usr/local/epics/Extensions

  - EPICS Archive Server
    - Archiver Appliance
      - /usr/local/epics/Config
      - /usr/local/epics/Extensions

  - EPICS Alarm Server
    - Tomcat/Kafka/Alarms
      - /usr/local/epics/Config
      - /usr/local/epics/Extensions
Motivation

• Historically, Fermilab is not an EPICS site
• PIP-II will use EPICS as its control system framework
• Treating EPICS deployment as a green field to simplify management of large control system
• Standard EPICS infrastructure to simplify developing IOCs for new developers
  - Developers start from template IOC
  - Template IOC has minimal basic functionality required of all FNAL IOCs
• Standard deployment for ease of Controls team debugging IOCs
• Server side – EPICS IOCs
  - Implement modern computing practices – git, CI/CD
  - Present LLRF code structure does not fit within our build scheme
• Client side
  - Update GUIs
  - Provide Save & Restore functionality
  - Provide data archiver
  - Provide alarms
  - Provide channel finder
• Strong collaboration, any changes in structure must remain in line with LLRF collaboration
DOE LLRF Collaboration

LLRF Teams from FNAL, JLab, SLAC and LBNL have been collaborating for the past 6 years in the context of LCLS-II and now PIP-II

Successful collaboration for LCLS-II and after 6 years we continue to want to work together
SLAC/LBNL Existing Software

- SLAC/LBNL software is stable and mature – see Carlos’ talk
- Used at SLAC, LBNL, FNAL, and JLab
- Code is well documented
- Code base is appropriately versioned in git
- Developed over long time, features added as needed, using different standards
  - Code is downloaded as two monoliths
  - Uses specific (older) version of EPICS base => cannot take advantage of EPICS7 features:
    - PV access – faster/higher bandwidth network protocol
    - Structured data
  - Scripts and configuration files in a variety of sub-directories
  - Structure presently incompatible with Fermilab deployment model

- Proposed code restructuring considerations:
  - Discussed with Sonya Hoobler (SLAC), to restructure code to match Fermilab deployment scheme
  - Will create a team of stake-holders from different labs to develop plan
  - It is critical to maintain compatibility between labs
  - Restructure location of software, *not* change software
  - Existing code makes use of environment variables which will simplify restructuring
  - Include modernization of existing GUIs

- Have built existing software using FNAL structure – still to be tested on teststand
Client Side Support

• Fermilab Controls team will support the following clients:
  - Archiver Appliance – in place and used at the end of PIP2IT
  - Kafka Alarm server – in progress
  - Channel Finder – in progress
  - Save & Restore – in place and used for much of PIP2IT
  - Convert edm → Phoebus – in progress
    • Significant project as there are >100 screens
    • Phoebus has conversion capability, but not all features are automatically ported

---

**Significant Project**

- Conversion of edm to Phoebus
- Project involves >100 screens
- Phoebus has conversion capability but not all features automatically ported
Future Considerations

- Build EPICS infrastructure for SoM/SoC architectures/platforms
- FPGA programmer does the hard work of programming FPGA and providing API
- It is not the job of the EPICS developer to define FPGA requirements
- Define requirements for API:
  - All large waveforms read directly from shared memory of FPGA and loaded to EPICS waveforms
  - Set value functions
  - Read value functions
- EPICS IOC will then call these functions from FPGA programmer provided share-object library and populate process variables (PVs in EPICS-speak)
- IOC will run on arm processor on SoM/SoC
  - IOCs are lightweight; i.e. they use little system resources
  - API also handles transport layer to communicate with the code
    - Must ensure to remain compatible with existing external server
- Presently have EPICS builds for: TerasicSoc (Cyclone V) and Achilles (Arria 10)
- Will add Xilinx based SoMs: MiniZed (Zynq 7Z007S), ZedBoard (Zynq-7000), TE0803 (Zynq UltraScale)
Summary

- Standard SLAC/LBNL implementations are running at Fermilab in CMTS, PIP2IT, and LLRF test stand
- Strategy for creating a robust EPICS deployment at Fermilab is well underway
- Strategy for modifying existing SLAC/LBNL code structure to fit a more
- Updating clients for PIP-II Teststand in progress