Fermilab LLRF Software

Architecture and Development

Paul W. Joireman
Fermi National Accelerator Lab
University Research Associates, Inc.

System Metrics

	System			
Metric	MI/MR	TV	RR	HEL
Year	1995	1996	1998	1992
VXI Cards	12	7	11	4
Slot 0*	10 (20)	7 (15)	7 (19)	1 (3)
DSP C/A*	3 / 10	0.6 / 4	0.5 / 7	-
Devices	381	191	390	50

* Thousands of lines of code, comments—not included here—double this number

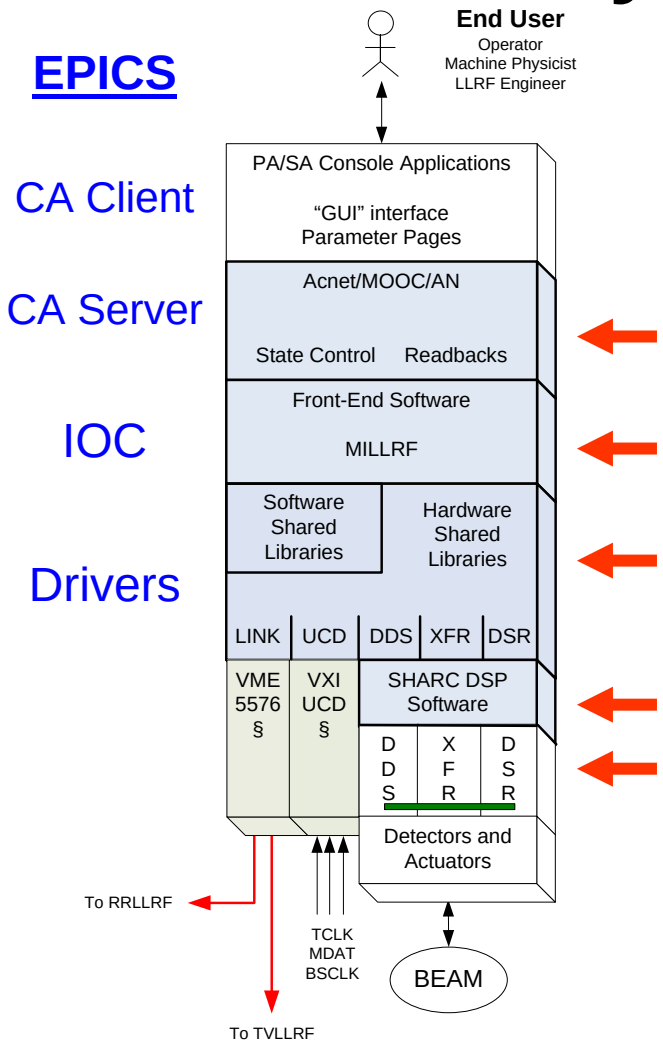
Unchanging Requirements

- **Reliability** – system must provide consistent results during continuous operation for a period of several weeks to many months.
- **Maintainability** – any required maintenance should be straightforward and transparent.
- **Adaptability** – the system must quickly and easily incorporate new requirements.
- **Usability** – the system should provide an intuitive and usable interface for a variety of end-users.

Managing Complexity

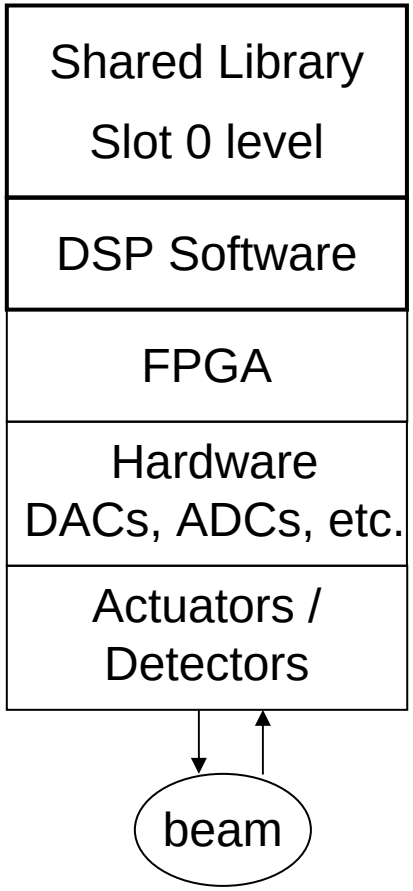
- Goals
  - Partition system tasks down to a manageable level
  - Maintain a flexible design and implementation
- Modularity and commonality
  - Modules
    - Strong Cohesion and Loose Coupling
    - Information Hiding/Encapsulation – hide the details
    - Benefit – increased reliability, maintainability and adaptability
  - Hardware modules – difference and commonality
  - Software modules – object-based methodology
  - Software patterns – solutions to a common problems

LLRF System Architecture



- AcNET Interface – requestor
- Front-end software - director
  - User interface
  - Scenario execution
- Shared Libraries – workers
  - Software – logger, event, cog,…
  - Hardware – link, VXI-UCD, VXI
- DSP Software – serfs
  - Patterns
- VXI Module Hardware
  - 40 MHz AD-SHARC™ DSP
  - ADCs, DACs, NCOs, Link ports

VXI Module Software



- DSP Software Patterns
  - Vector Interrupts
  - Link Port Communication
  - Scheduler
  - Fast Time Plot Variables (PV)
- Object Based Methodology
  - Construction
  - Configuration
  - Information

Software Modules

- AN (Channel Access Server)
  - Acnet interface, callback mechanism
- Event
  - Observer pattern or Publish-Subscribe model
- Logger
  - Message queue used to log error messages
  - Different message levels, information content
- Cog
  - manages cogging beam to specific RF buckets
  - Details of cogging hidden, interface clear

Front-End Software

- The “application program” or *.exe file (VxWorks)
  - Provides set of services or tools to end-user.
  - Coordinator of tasks within front-end by combining services provided by different modules.
  - Responds to IRQs, TRIGs or external requests for data.
  - Error detection, logging and recovery.
- Basic Control System Interface – devices (PV)
- Advanced Control Interface – I6/R6
  - Scenarios - UseManager, Multi-timer


VMS PC:I6 LLRF VXI CONTROL<NoSets>
I6 VXI LLRF Novice Mode (most restrictive) 16-SEP-05 13:02:31 ◆Pgm_Tools◆
SEQUENCE TABLE VIEWER
CURVES
ARRAYS
REQUIRED DATA
Sequence Table View
HrdwrID: [MILLRF] *Send To Hardware *Refresh
MI State: [21 Numi 6 batches] Current state: 21
ROW TYPE SIGNAL MESSAGE DATUM1 DATUM2 DATUM3 DATUM4
0 Event AnyReset EnergyStepToFset 52811400
1 Continue EnergyArmATC 0 3048000
2 Continue XfrSyncMItoBooster 36 80 -30
3 Continue SetPhisFrontEndAtten 40 14
4 Continue V588Apg(t)Curve All On .0014
5 Continue Scope Trigger 1
6 Event BooPInject QdotfbOn 0 8Gev LPF
7 Continue Delay usec 3000
8 Continue XfrSyncBoosterToMI 122 70 0
9 Event BooPInject Delay usec 2000
10 Continue XfrSyncBoosterToMI 208 85 0
11 Event BooPInject Delay usec 2000
12 Continue XfrSyncBoosterToMI 294 60 0
13 Event BooPInject Delay usec 2000
14 Continue XfrSyncBoosterToMI 380 90 0
15 Event BooPInject Delay usec 2000
16 Continue XfrSyncBoosterToMI 466 100 0
17 Event BooPInject
18 Delay 0.4620000124 EnergyQrpfb -10 -20000 .1 All DSR
19 Delay 1.1499999762 EnergyQrpfb -2 -2000 .1 All DSR
20 Delay 1.1599999666 QdotfbOn 0 120Gev LPF
21 Delay 1.1699999571 EnergyRampToFset 53103480 .02
22 Delay 1.2344300747 RfH588StationControl All On 1 Enable
23 Continue Delay usec 1840
24 Continue RfH588StationControl All On 1 Enable
25 Event EndCycle
26
27
28
29
SEQUENCE : Rece
SEQUENCE : Requ
SEQUENCE : Rele
SEQUENCE : Rece
SEQUENCE : Requ
LLRF: initializ
PGM: Signal hel
Required Data
BEAM INJECTOR TARGET Qrpfb Curve LOG Reports BATCH V28PGM h588 Curves
Protons Booster NuMI 0 Minimum Full 0 I6 Expert2
1:7 of 9

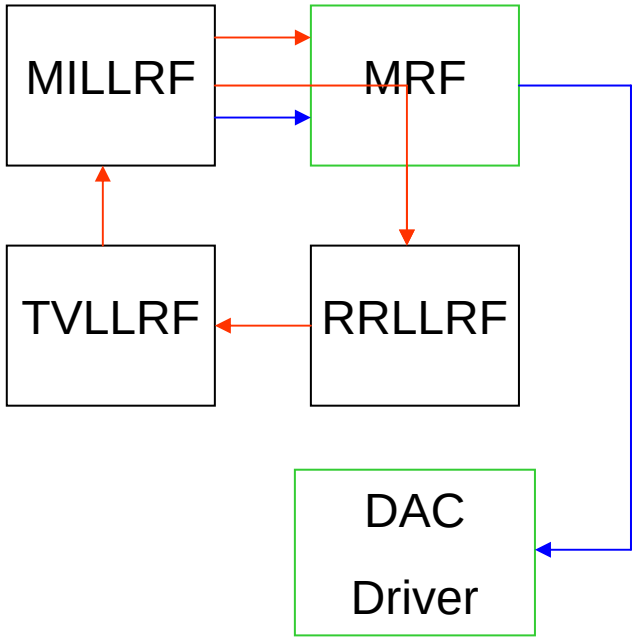
Software Process

- Three release levels
  - PRODUCTION – operational code
  - TEST – new features for test and deployment
  - DEVELOPMENT – new code development
- Lab system – separate VXI crate
  - Mirror of the operational system
  - Used test all changes prior to release
- Code repository – CVS
- Coding standards – common style

A Case Study – MRF

- Middle Level RF (MRF) Requirements
  - Global regulation of HLRF cavity amplitude and phase moved from analog to digital.
  - State mapped curve output from DAC Driver.
- Options
  - Implement changes on MILLRF system
    - Few slots available
    - Computational load already high
  - Implement changes on a new VXI system
    - Couple system to existing MILLRF system

MRF System



- MRF node on optical link
  - Link job server for MILLRF
- Data link MILLRF → MRF
  - 100 kHz real-time data
- Data link MRF → DAC
  - State-mapped curves
- Implement MRF functions
  - Hardware – existing modules
  - Software
    - Existing modules and patterns
    - DSP Software

MRF Results

- Integration
  - Seamless and reasonably transparent (to end-user)
  - Facilitated by loosely coupled support libraries
- Work required
  - Time frame: 6 months
  - Effort breakdown
    - VXI hardware (5 %) – existing modules
    - Front-end software (15 %) – reuse of software modules
    - DSP Software (30 %) – reuse of DSP patterns
    - DAC Hardware/FPGA programming (50 %) – new design
- Results
  - +/- 0.5 % amplitude variation in RF cavity voltage
  - +/- 0.5 degrees phase variation

Summary

- Remember the unchanging requirements
  - Initial Design
  - Implementation
  - Upgrades/Maintenance
- Modularize to manage complexity and meet unchanging requirements
- Build flexibility into system from the first
  - Interchangeable components
  - Reusable modules