BPM Software Architecture and FCOM Integration

Design Review

Till Straumann

11/10/2009

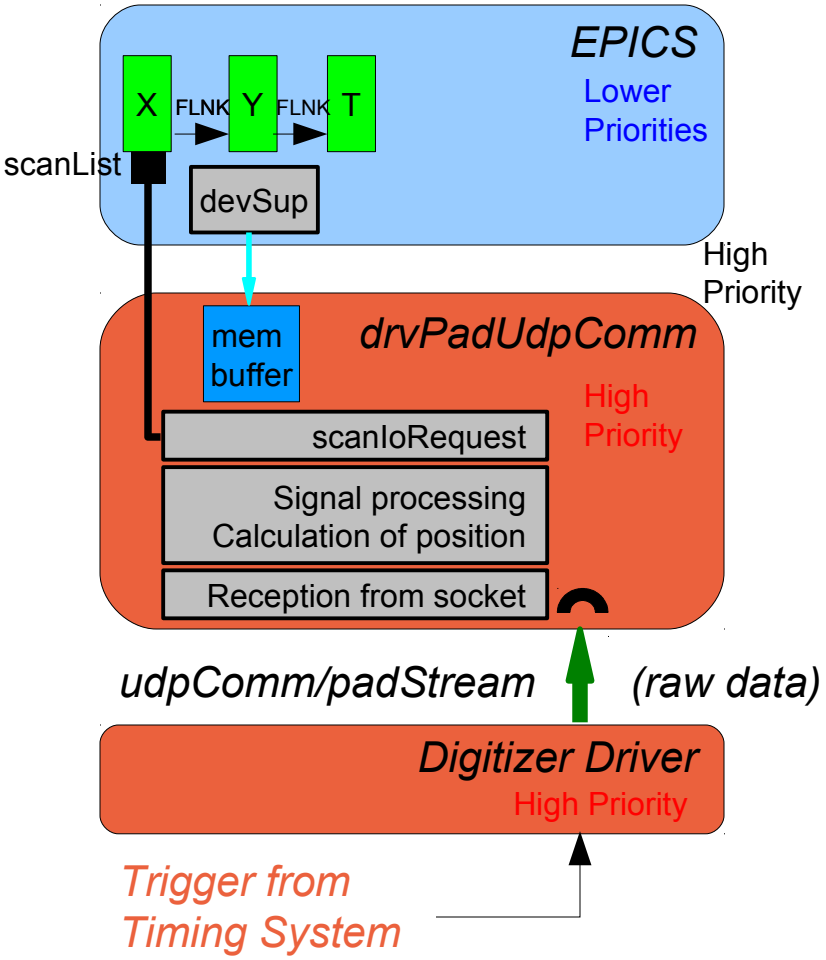
Overview

Controls

- Software architecture
- Plan for adding FCOM “output”
- Required changes and additions
- Estimated effort
- Plan for migrating PAD communication from using broadcast to using multicast
- Estimated effort
- Summary

Architecture Overview

Controls



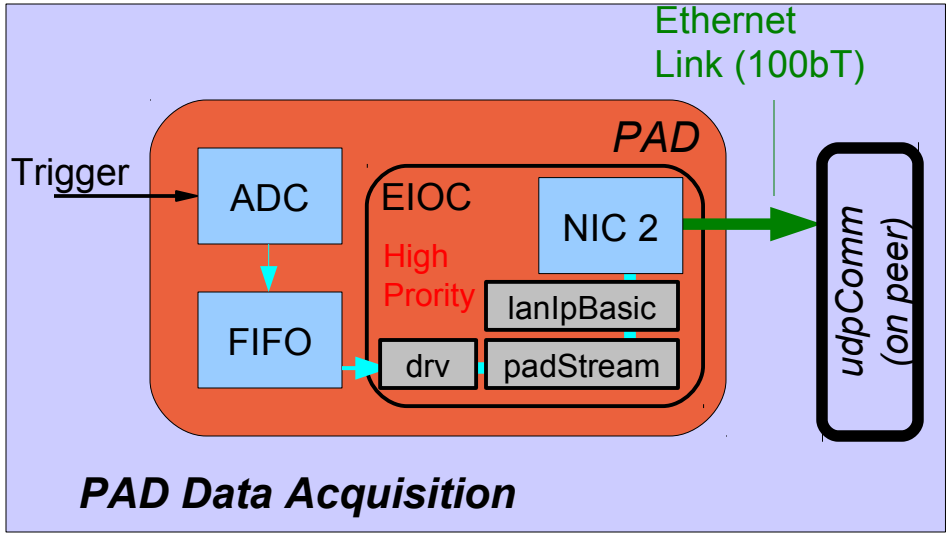
From bottom up:

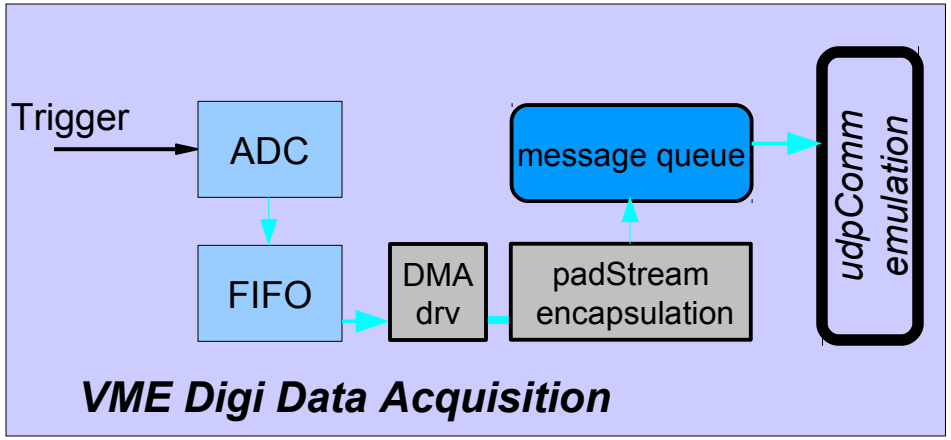
- Event driven; triggered by timing system (hardware acquisition trigger → ADC IRQ).
- ADC driver packs raw data in 'padStream' UDP message.
- drvPadUdpComm reads raw data from socket, performs signal processing, calculates position and deposits values in memory buffer.

  When done, drvPadUdpComm notifies EPICS by means of `scanIoRequest()`.
- X/Y/TMIT record chain, when scanned, retrieves data from memory buffer (only shown for Y).

Digitizer Drivers

Controls



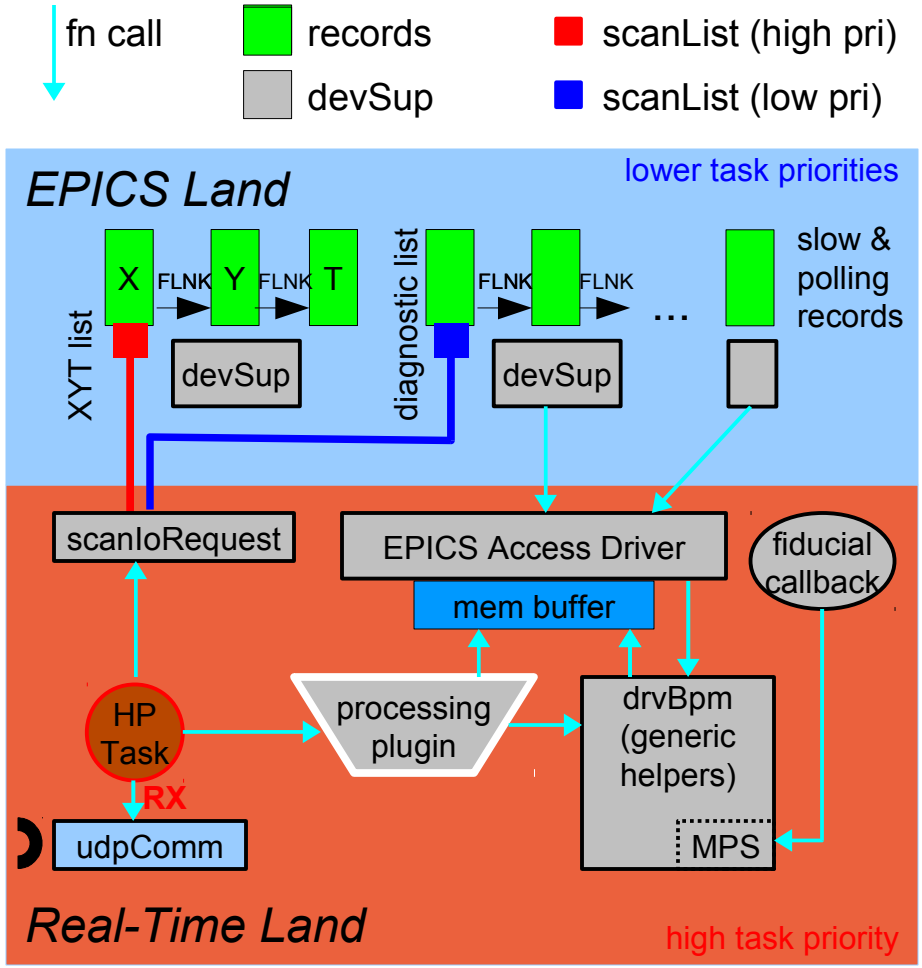


- Modular design presenting unique interface.
- Raw waveforms are shipped as UDP/"padStream" protocol packets.
- "padStream" adds meta-data header: element count, endian-ness, flavor (calibration/beam data), timestamp, etc.
- On PAD a high-priority task reads ADC data and ships "padStream" packet via dedicated ethernet link (using dedicated IP/UDP mini-stack "lanIpBasic").

  EIOC also runs EPICS, controlling front-end attenuators + diagnostics (lower priority).
- VME digi driver uses DMA and implements udpComm emulation.
- Higher-level SW has no knowledge of type of digitizer used.

BPM Processing

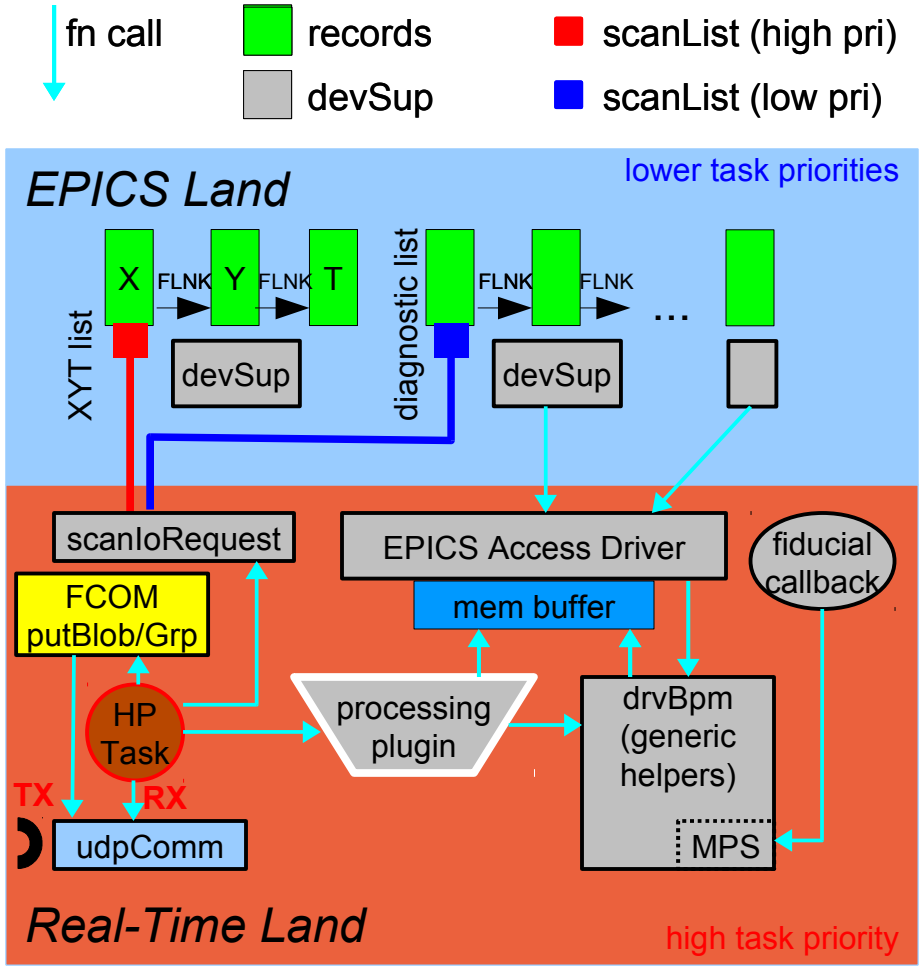
Controls



- Only main processing shown (w/o ADC driver).
- High priority task receives raw data from udpComm layer. Processes data via "plugin" module which handles stripline vs. cavity algorithms.
  Generic routines are provided by 'drvBpm'. "Cooked positions" and diagnostic data are deposited in memory buffer.
- High (X,Y,TMIT chain) and low priority scan lists (diagnostics) + passive and polling records in EPICS land.
- EPICS fetches data from buffer via devSup/driver.
- Fiducial callback scheduled by timing system performs MPS logic.

Adding FCOM Support

Controls



- Addition of FCOM is relatively straightforward: after positions are calculated, HP task assembles a blob and adds it to a group.
  When last BPM is done, send group out.

  Caveat: if one BPM is down *nothing* gets sent until bad BPM is taken off-line (button on diagnostic display).
- Problem with VME digitizer: these systems now require *real* udpComm/lanIpBasic which collides with emulation that is currently used.

  Possible solutions
  - Give up binary compatibility and link cavity BPMs with emulation (simple but ugly).
  - Add indirection through new function-pointer table (more elegant).

Required Work (FCOM/BPM)

Controls

- *Administrative (someone else + networking):*
  - Assign FCOM IDs (same GID for all BPMs hosted on a single IOC).
  - Assign DNS-names/IP addresses for FCOM nodes.
  - Implement lookup service (DNS?) for 2nd NIC IP.
  - Implement lookup for FCOM IDs.
- *Interface:*
  - Define blob format for data, timestamp and status.
- *Software:*
  - Use lookup service(s); initialize lanIpBasic + FCOM.
  - Resolve conflict of udpComm emulation (VME digi) with FCOM.
  - Post BPM readings to FCOM.
- Estimated effort: ~4 weeks.

Migrating PAD Communication

Controls

- Currently, VME IOC broadcasts to PADs (on private network). PADs 'learn' VME IOC's IP & MAC address and send unicast back.
- Not viable if formerly “private” LAN joins the feedback-network (FNET).
- Proposal: Use dedicated MC group for each VME/master IOC
  - Master IOC posts to MC group
  - PADs subscribe to group
  - PADs receive MC messages and send unicast back
- Configuration issue: Current implementation could use same ('hardcoded') IP subnet address for all systems.
  New implementation requires
  - Unique IP addresses for all nodes (master IOCs and PADs)
  - Unique MC group for each master IOC
  - Master IOC and PADs must now “know” or “learn” to which MC group they belong

Required Work (MC Migration)

Controls

- *Administrative (resolution requires working with network group):*
  - Assign IP addresses for all 2rd NICs (VME IOCs and PADs) on FNET.
  - Assign MC address for each VME IOC (on FNET). ***Must not*** overlap MC addresses used by FCOM.
  - Add all these addresses to DNS.
  - Implement some form of convention so that master IOCs and PADs can find out what MC group they must subscribe to.
    (E.g., let DNS names "`<eioc-base-name>-bpmgrp`" and "`<ioc-base-name>-bpmgrp`" map to MC address.)
- *Software:*
  - Initialization of lanIpBasic needs to lookup addresses.
  - Lookup of MC group address during initialization.
  - Subscribe (PAD) and post (VME IOC), respectively, to MC group.
  - PADs may block BCST at 2rd NIC to avoid flooding.
- Estimated effort: ~2 weeks.

Summary

Controls

- Required changes are modular and relatively small.
- However, there are some “nasty” details to be addressed.
- More administrative overhead; configuration spread (from startup script → name + boot servers and their databases).
- Estimated work load ~6 weeks.